The Dissertation Committee for Arindam Banerjee
certifies that this is the approved version of the following dissertation:

# Scalable Clustering Algorithms

Committee:

Joydeep Ghosh, Supervisor

Adnan Aziz

Ross Baldick

Inderjit Dhillon

Gustavo de Veciana

# Scalable Clustering Algorithms

by

**Arindam Banerjee, M.Tech, B.E.**

**Dissertation**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

**Doctor of Philosophy**

**The University of Texas at Austin**

August 2005

To My Parents

# Acknowledgments

I would like to thank a number of people who contributed in their own unique ways in making my experience as a graduate student very rewarding. First, I would like to thank my advisor Prof. Joydeep Ghosh for his support and guidance over the years. Prof. Ghosh always showed great faith in my capabilities and have allowed me to work quite independently at times, while providing invaluable guidance when necessary. His friendly accessibility and constant encouragement has been one of the key factors that has helped me mature as a researcher.

I am grateful to Prof. Inderjit Dhillon for his enthusiastic encouragement and often pushing me to think more deeply about certain research issues. I have learnt various other aspects of being a good researcher from him, including technical writing as well as presentation skills. I would also like to thank Prof. Ray Mooney for introducing me to Machine Learning, and giving me the opportunity to collaborate with him on several occasions. I want to thank my committee members for inputs at various levels of my thesis work.

I would specially like to thank Srujana Merugu for being my most active collaborator and great friend. I will sorely miss the exchange of ideas and the intense collaboration we have developed, spending hours in front of the white board while munching on "public candy". I want to thank all the members of LANS for being patient listeners of my half-baked ideas, giving constructive feedback, and overall being such nice friends. Special thanks go to our student system administrators - Sreangsu Acharyya and Chase Krumpelman, for doing a terrific job.

A large part of my value system for judging as well as performing good research was a result of my close interactions with Mama (Sandip Ray). Be it overnight discussions at our apartment, or over a 'hammerhead' at our favorite coffee-shop Metro, debates and discussions with Mama have been always enlightening.

A Big 'Thank You' goes to my very close friends at our own little IEEE — the institute for excessive eating engineers. Thanks to all of you: Anirban, Sandhitsu, Siddhartha, Sreangsu and Sujata for being such great friends. I will always cherish the awesome IEEE conferences. I would like to thank my awesome house-mates at various times: Sreangsu, Mama, CV and Manoj for tolerating me gracefully. I want to thank Austin and its people for providing me a great home away from home. In particular, I would like to thank my friends at UTBC, our great hang-outs at Posse-East with Hari, CV, Dwip-da, and the rest of the gang. I consider myself lucky to have Sugato Basu as a friend and collaborator. Special thanks goes to Arjun Khanna for the Neonyoyo and the Dog & Duck sessions.

I am indebted to IBM for supporting my research in academic years 2003-2004 and 2004-2005 through a fellowship. Working with people in the Data Analytics Research group at IBM has given the much needed practical insight in my research. I also want to thank Xin Guo and John Langford, whom I met at IBM and who eventually became my collaborator and good friends.

Finally, I want to thank Ma, Baba, Dada and Tiklu-di for their unwavering support and encouragement and for being such great friends all along.

Arindam Banerjee

*The University of Texas at Austin*

*August 2005*

# Scalable Clustering Algorithms

Publication No. _____

Arindam Banerjee, Ph.D.
The University of Texas at Austin, 2005

Supervisor: Joydeep Ghosh

Scalable clustering algorithms that can work with a wide variety of distance measures and also incorporate application specific requirements are critically important for modern day data analysis and predictive modeling. In this thesis, we propose and analyze a large class of such algorithms, evaluate their performance on benchmark datasets and investigate theoretical connections of the proposed algorithms to lossy compression and stochastic prediction.

First, a wide variety of popular centroid based clustering algorithms are unified using a large class of distance measures known as Bregman divergences. We present both hard and soft-clustering algorithms using Bregman divergences. By establishing a bijection between regular exponential family distributions and regular Bregman divergences, we note that Bregman soft clustering algorithms are

equivalent to learning mixtures of exponential family distributions, but can be computationally more efficient in practice. We also design algorithms for clustering directional data that generate balanced clusters, i.e., clusters of comparable sizes, a desirable property in certain practical applications. Experimental results show that such algorithms perform well for high-dimensional problems such as text clustering.

A general framework for scaling up balanced clustering algorithms is then proposed. The framework is applicable to all the algorithms presented in this thesis as well as a wide variety of other algorithms. Extensive experimental results on benchmark datasets are provided to establish the efficacy of the proposed framework. Further, we propose a new method for evaluation and model selection for clustering that can be applied to practically any clustering algorithm. The method is applicable in a transductive setting and measures the predictive accuracy of a clustering algorithm.

A detailed analysis of the connections of rate distortion theory to the proposed clustering algorithms, in particular the Bregman clustering algorithms, is also presented. In the process, we establish some key theoretical results in rate distortion theory for Bregman divergences, special cases of which has been studied in the literature using squared Euclidean distance. Also, we generalize a widely known result in stochastic prediction by establishing that the conditional expectation is the optimal predictor of a random variable if and only if the prediction error is measured by a Bregman divergence. This results explains the fundamental reason behind the efficiency of the Bregman clustering algorithms.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Clustering can be defined as the process of organizing a collection of patterns into groups or clusters whose members are similar in some way. As a branch of statistics, cluster analysis has been studied extensively for many years, focusing on both similarity-based as well as distance-based clustering [JD88, ELL80]. Clustering has also been studied in the field of machine learning as a type of unsupervised learning because it does not rely on predefined class-labeled training examples [DHS00]. However, efforts to perform effective and efficient clustering on large datasets only started in recent years with the emergence of data mining.

There are two major approaches to clustering: *generative* and *discriminative*. In a generative approach, the patterns are assumed to have been generated by a probabilistic pattern generation process that is dependent on certain parameters. Quite often there is a well-defined mapping between these parameters and the data clusters, though in some cases there is no such explicit mapping. The corresponding clustering algorithm tries to make the best estimate of the parameters and obtains the data clusters using these estimates. Such a framework corresponds to the model-based or parametric approaches in classical pattern recognition. In a discriminative approach, no assumption is made regarding the source or method of generation

of the patterns. However, it is assumed that the patterns exist in a space that has a well-defined distance or similarity measure between any pair of patterns in that space. The patterns are placed into separate groups so that patterns within a cluster have high similarity with one another but are dissimilar to patterns in other clusters. This roughly corresponds to the non-parametric approaches in classical pattern recognition.

Clustering comes in two flavors: *hard* and *soft*. In hard clustering, every data point uniquely belongs to one cluster. Thus, a clustering algorithm effectively partitions the data into disjoint sets based on their cluster assignment. Hard clustering algorithms can be either generative or discriminative. In soft clustering, data points have a probability of being assigned to all the clusters. Soft clustering is naturally more closely associated with generative models. Further, hard clustering can be viewed as a special case of soft clustering where the probabilities are either 1 or 0.

## 1.1 Why Scalable Clustering Algorithms?

With the development of computational and information technology, a plethora of digital data in various forms, e.g., content and link information from the web, hyperspectral data from satellites, transaction data from businesses, microarray gene expression data from computational biology, stock and currency trading data from financial markets, are now readily available. We look at a few examples to understand the scale of the current datasets where data analysis and clustering is routinely performed:

1. A publicly available benchmark dataset sampled from 20 Usenet newsgroups, has 1000 English text messages from each newsgroup [N20]. Using the "bag of words" model for text analysis, the dimensionality of the data is around 25,000, which is the size of the dictionary used to model the data. Since each newsgroup post uses only a small number of words from the entire dictionary,

the data is very sparse. Thus, the data has 20,000 very sparse data points in a 25,000 dimensional space.

2. Another interesting dataset is based on National Science Foundation (NSF) grant proposals [NSF]. The data consists of abstracts of more than 130,000 grant proposals that were given funding over a period of 14 years, from 1990 to 2003. With a "bag of words" model, the dimensionality of the data is around 45,000. The data is sparse since each abstract uses only a very small fraction of the words in the entire dictionary.

3. Among the public domain movie recommendation data available, the popular `Eachmovie` dataset, collected by HP/Compaq research for over 18 months from 1995 to 1997, consisted of 2,811,983 ratings entered by 72,916 users for 1628 different movies [Mov].

Such recent benchmark datasets are clearly orders of magnitude larger than the classical predictive modeling datasets such as Iris, which contains 150 data points in 4 dimensions. However, certain real world datasets are orders of magnitude larger than even these "large" benchmarks. We consider a few examples:

1. The actual Usenet has close to 950 top level categories and around 55,000 newsgroups. The total number of messages in all newsgroups cached over several years is over a billion postings.

2. Web search is a domain that has motivated significant recent research in data analysis and clustering. The popular search engine `Google` currently indexes over 8 billion webpages.

3. The Dun & Bradstreet (D&B) database is a widely used resource where data analysis techniques are frequently applied. The database contains relevant information on more than a million American and Canadian companies, as well as an equally large number of international companies.

Data clustering has been applied to various degrees on all the above examples and many others.

Since clustering often forms the first-stage analysis before applying other data mining techniques, it is important in many (perhaps most) domains that the clustering algorithm be fast and must be able to handle very large datasets. In fact, clustering is often used as a pruning mechanism so that un-interesting and obvious clusters as well as outliers can be discarded before proceeding with further analysis. In other situations, such as clustering of genes using micro-array data, clustering is the critically important step in the analysis and getting good quality clusters is important. Such considerations motivate the study of both fast clustering algorithms with provable properties as well as methods for scaling up existing clustering algorithms.

In addition to the requirement of being scalable, modern day problems often pose other requirements or constraints on clustering models and algorithms, that need to be considered for the algorithms to be useful in practice:

1. Unlike classical datasets such as Iris, which were low-dimensional and well-separated according to Euclidean distance, most of the modern day datasets have diverse characteristics that violate such assumptions. Often they reside in spaces where Euclidean distance is not meaningful. Hence, it is desirable to have clustering algorithms that are applicable for a large class of distance measures. Then, for a given application, the practical task will be only to choose an appropriate distance function from that class.

2. Many practical clustering tasks require that the clusters generated be of comparable sizes. For example, a direct marketing campaign often starts with segmenting customers into groups of roughly equal size or equal estimated revenue generation, (based on, say, market basket analysis, or purchasing behavior at a web site), so that the same number of sales teams or marketing

dollars can be allocated to each segment. Even when such a balancing requirement is not there, it is often desirable to introduce such constraints to prevent empty or degenerate cluster formation.

3. The typical view of a dataset being a fixed flat file is not very realistic in many situations. In practice, datasets grow over time. For example, for a large retail store such as `Walmart` or online business such as `Amazon`, every transaction is recorded and the size of the purchase data grows every day. Hence, it is desirable to have online clustering algorithms that are applicable to ever growing data without having to store everything explicitly.

In this thesis, we focus on developing scalable clustering algorithms under one or more of the requirements outlined above.

## 1.2 Overview

Chapters 3-7 are devoted to the development and analysis of efficient clustering algorithms. Chapters 3 and 4 present clustering algorithms that can use a large class of distance measures known as Bregman divergences. Chapter 5 focuses on efficient clustering of directional data, such as certain text datasets, while generating approximately balanced clusters. In Chapter 6, we present a general framework for scaling up clustering algorithms. Evaluation and model selection is an important issue in clustering and we propose a new method in Chapter 7. Chapters 8 and 9 present theoretical analyses showing connections of our work, in particular the Bregman clustering algorithms, to rate distortion theory and stochastic prediction. The flow of the thesis is shown in Figure 1.1.

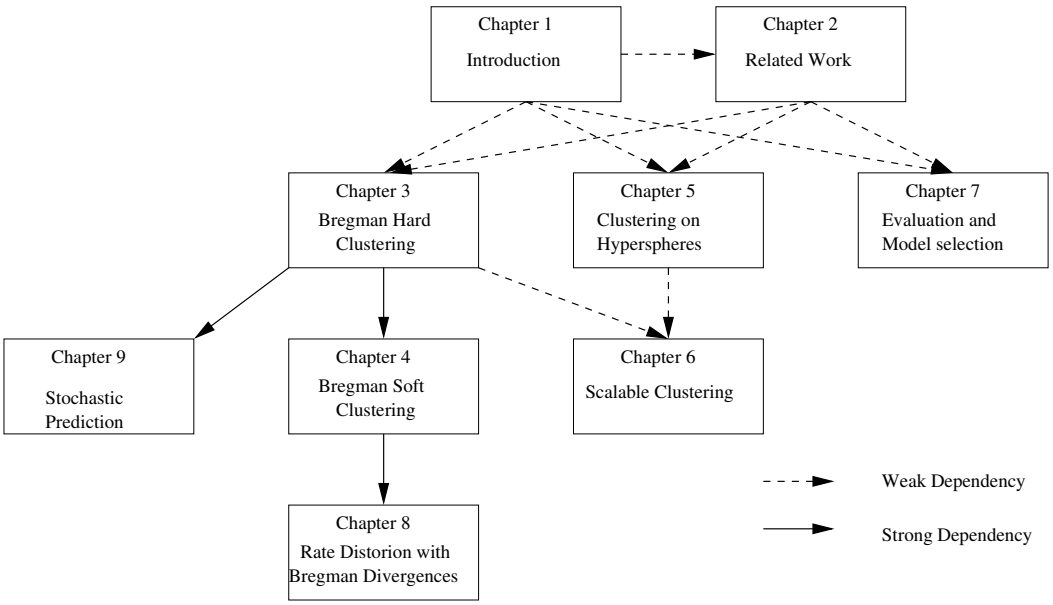


Figure 1.1: Flow of the thesis

### 1.2.1 Hard Clustering with Bregman Divergences

In Chapter 3, we present a class of hard clustering algorithms based on a very general class of distortion functions called Bregman divergences. Since there is a Bregman divergence corresponding to every strictly convex and differentiable convex function, the resulting class of algorithms is quite general. Further, it includes several popular algorithms such as kmeans, information-theoretic clustering, LBG clustering etc., as special cases for particular choices of convex functions.

### 1.2.2 Soft Clustering with Bregman Divergences

In Chapter 4, we present a class of soft-clustering algorithms based on Bregman divergences. In the process, we prove a fundamental connection between Bregman divergences and the exponential family of distributions, which include almost all popular parametric distributions such as Gaussians, Bernoulli, multinomial, Poisson, etc. Further, the connection also implies that our soft-clustering algorithms give an

efficient way to fit mixtures of exponential family distributions to observed data.

### 1.2.3 Clustering on the Hypersphere

It has been observed that $L_2$ normalization of data often forms a better representation in several important domains, such as information retrieval and recommender systems. In Chapter 5, we present batch as well as online algorithms for clustering data that is $L_2$ normalized and hence lie on the unit hypersphere.

### 1.2.4 Scalable Clustering with Balancing Constraints

In many practical scenarios, the requirement is to get a scalable clustering algorithm with a pre-specified minimum number of points per cluster. Chapter 6 presents a general framework for obtaining scalable clustering algorithms in the presence of such balancing constraints. The framework is applicable to a large class of clustering algorithms including all algorithms discussed in the earlier chapters.

### 1.2.5 Evaluation and Model Selection for Clustering

Evaluation and model selection, in particular, selecting the number of clusters is a critical issue in clustering. In Chapter 7, we present an objective evaluation selection criterion for clustering based on its predictive performance. The criterion also gives a way of model selection in that models that have low error-rate bounds on future data are preferred.

### 1.2.6 Rate Distortion with Bregman Divergences

Rate distortion theory studies the fundamental limits of lossy compression of a stochastic source. In Chapter 8, we present results in rate distortion theory when distortion is measured by a Bregman divergence. In particular, we show that the rate distortion function can be either obtained analytically or computationally. The

computational algorithm is exactly the same as the corresponding Bregman soft clustering algorithm. The analysis gives a theoretical understanding of what the clustering algorithms are trying to achieve: optimal lossy compression.

### 1.2.7 Optimal Stochastic Prediction

In Chapter 9, we study optimal stochastic prediction when error in prediction is measured using a Bregman divergence. A widely used result in probability theory states that the conditional expectation is the least square predictor. We show that the conditional expectation is the optimal predictor if and only if loss is measured by a Bregman divergence. The well known result in probability theory follows as a special case of our result since squared Euclidean distance is a Bregman divergence. Further, a key property that makes our proposed Bregman clustering algorithms efficient and scalable is also due to a special case of this result.

## 1.3 Notation

A word about the notation: bold faced variables, e.g., $\mathbf{x}, \boldsymbol{\mu}$, are used to represent vectors. Sets are represented by calligraphic upper-case alphabets, e.g., $\mathcal{X}, \mathcal{Y}$. Random variables are represented by upper-case alphabets, e.g., $X, Y$. The symbols $\mathbb{R}, \mathbb{Z}$ and $\mathbb{R}^d$ denote the set of reals, the set of integers and the $d$-dimensional real vector space respectively. Further, $\mathbb{R}_+$ and $\mathbb{R}_{++}$ denote the set of non-negative and positive real numbers. For $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, $\|\mathbf{x}\|$ denotes the $L_2$ norm, and $\langle \mathbf{x}, \mathbf{y} \rangle$ denotes the inner product. Unless otherwise mentioned, log will represent the natural logarithm. Probability density functions (with respect to the Lebesgue or the counting measure) are denoted by lower case alphabets such as $p, q$. If a random variable $X$ is distributed according to $\nu$, expectation of functions of $X$ are denoted by $E_X[\cdot]$ or by $E_\nu[\cdot]$ when the random variable is clear from the context. The interior, relative interior, boundary, closure and closed convex hull of a set $\mathcal{X}$ are denoted by $\text{int}(\mathcal{X})$,

$\mathrm{ri}(\mathcal{X})$, $\mathrm{bd}(\mathcal{X})$, $\mathrm{cl}(\mathcal{X})$ and $\mathrm{co}(\mathcal{X})$ respectively. The effective domain of a function $f$, i.e., set of all $x$ such that $f(x) < +\infty$ is denoted by $\mathrm{dom}(f)$ while the range is denoted by $\mathrm{range}(f)$. The inverse of a function $f$, when well-defined, is denoted by $f^{-1}$.

# Chapter 2

# Related Work

This chapter gives a brief survey of previous work on data clustering and related literature. The algorithms and techniques described in this survey are from both the classical as well as the more recent data-mining oriented pattern recognition community.

This chapter has six sections, each one appropriate for a set of subsequent chapters. In section 2.1, a survey of existing well-known clustering techniques is presented. This a summary of the basic ideas as well as the state of the art in clustering, although it is far from comprehensive. In section 2.3, we review how a large class of distortion functions called Bregman divergences have been used in the machine learning literature. Section 2.2 summarizes various methods that have been proposed in the data mining literature for scaling up clustering algorithms. In section 2.4, we summarize standard evaluation and model-selection techniques for clustering, which remains a widely debated issue to date. We briefly review rate distortion theory in section 2.5. Finally, we discuss the basics of stochastic prediction in section 2.6.

## 2.1 Clustering techniques: A brief survey

Depending on the assumptions made while formulating the clustering problem as well as the way the algorithm handles the data, clustering algorithms are typically roughly divided into four major categories: *partitioning methods*, *hierarchical methods*, *density-based methods*, and *graph-based methods*. The basic model of clustering used in each of these methods will be discussed as the algorithms are presented. In fact, two different algorithms under the same method may actually be based on the two different models of clustering. Most of these algorithms have been designed and nurtured to work on vector type data only and there is no direct way to make them handle sequences or other non-vector type data.

### 2.1.1 Partitioning methods

Partitioning algorithms for clustering have been popular long before the emergence of data mining. Given a set of $n$ objects in a $d$-dimensional space and the number of clusters $k$, a partitioning algorithm organizes the objects into $k$ clusters such that the total deviation of each object from its cluster representative is minimized. Typically, this approach follows the generative model for clustering.

This section presents a brief discussion on the three most well known partitioning methods: the *expectation maximization* (EM) algorithm (for mixture densities) [DLR77], the `KMeans` algorithm [Mac67, DHS00], and variants of the `KMedian` algorithm [JV01]. The three algorithms have different assumptions regarding the representation of the clusters and hence try to optimize different objective functions based on those assumptions. However, all of them essentially try to find $k$ cluster centers or distributions that best represent the observed data under certain assumptions. Finding the globally optimum $k$ centers or distributions is known to be NP-hard and these algorithms only guarantee a local optimum.

The generative model assumptions typically used for such algorithms is as

follows: the set of $n$ observed points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ actually come from a underlying parametric distribution which is a mixture of $k$ probability distributions so that for any observed point $\mathbf{x}$,

$$p(\mathbf{x}|\Gamma) = \sum_{h=1}^{k} \alpha_h p_h(\mathbf{x}|\theta_h) \tag{2.1}$$

where the parameter vector $\Gamma = (\alpha_1, \cdots, \alpha_k, \theta_1, \cdots, \theta_k)$ such that $\sum_{i=1}^k \alpha_i = 1$ and each $p_h$ is a density function parameterized by $\theta_h$. The most commonly used component density function is the $d$-dimensional Gaussian for which $\theta_h = (\mu_h, \Sigma_h)$, where $\mu_h$ is the mean and $\Sigma_h$ is the covariance of the Gaussian representing the $h$th component, and

$$p_h(x|\theta_h) = \frac{1}{(2\pi)^{d/2}|\Sigma_h|^{1/2}} e^{-\frac{1}{2}(x-\mu_h)^T \Sigma_h^{-1}(x-\mu_h)} \tag{2.2}$$

Each of these components are assumed to represent a cluster $C_h, h = 1, \cdots, k$ and each of the $n$ points are assumed to come from exactly one of the components and hence belong to the corresponding cluster. The clustering problem is to make the best possible estimate of the component densities under certain assumptions and assign a cluster label to each of the data points based on the most likely component that might have generated it. Let $X$ denote the random variable corresponding to the mixture distribution. Let us assume the existence of a random variable $Z$ that contain necessary information about the component generating any particular sample, i.e., the cluster assignment of any sample. Let $\mathcal{Z} = \{\mathrm{z}_i\}_{i=1}^n$ be the set of such assignments for the entire observed data $\mathcal{X}$. Assuming that the samples were drawn independently, the clustering algorithm tries to find the values of the model parameters so as to maximize the log-likelihood of the observed data $\mathcal{X}$, given by

$$\mathcal{L}(\mathcal{X}, \mathcal{Z}|\Gamma) = \sum_{i=1}^{n} \log p(\mathbf{x}_i, \mathrm{z}_i|\Gamma) \tag{2.3}$$

Note that unless $\mathcal{Z}$ is known, one cannot directly solve the maximum likelihood estimation problem. However, $\mathcal{Z}$ is not known in practice. The classical way to address this problem is to find the parameters so that the expected value of the log-likelihood is maximized where the expectation is computed over the conditional distribution $p(Z|X,\Gamma)$ [DLR77]. The expected log-likelihood is given by

$$E_{Z|X,\Gamma}[\mathcal{L}(\mathcal{X},\mathcal{Z}|\Gamma)] = \sum_{i=1}^{n} E_{Z|\mathbf{x}_i,\Gamma}[\log p(\mathbf{x}_i, Z|\Gamma)] \tag{2.4}$$

Again, this cannot be directly solved since $p(Z|X,\Gamma)$ is not known. This problem is solved by the Expectation Maximization technique that starts from an arbitrary choice of $\Theta$ and iteratively improves the estimates of $\Gamma$ and $p(Z|X,\Gamma)$ while trying to maximize the current estimate of the expected log-likelihood [DLR77]. The technique is guaranteed to converge to a local maximum of the expected log-likelihood.

Clustering algorithms based on EM differ from each other depending on what assumptions are made about the density components, the nature of $Z$ and how the conditional distribution $p(Z|X,\Theta)$ is computed. In EM for Mixture of Gaussians, the individual components are assumed to be Gaussians and EM is applied to get the clustering. In `KMeans`, it is further assumed that $\alpha_h = \frac{1}{k}$, $\Sigma_h = \epsilon\mathbb{I}$, where $\mathbb{I}$ is the identity matrix, and $\epsilon \to 0^+$ [KMN97]. `KMedian` is similar in flavor to `KMeans` but has the extra assumption that $\mu_h \in \mathcal{X}, h = 1, \cdots, k$. This makes `KMedian` an integer programming problem that the general EM framework cannot handle. Hence solving the `KMedian` problem requires a rather different approach [JV01].

### 2.1.2 Hierarchical methods

A hierarchical method creates a hierarchical decomposition of the given data objects forming a dendogram. The dendogram can be formed in two ways: *bottom-up* and *top-down*. The *bottom-up* approach, also called the *agglomerative* approach, starts

with each object forming a separate group. It successively merges the objects or groups that are closest according to some distance measure, until a termination condition is satisfied. The *top-down* approach, also called the *divisive* approach, starts with all the objects in the same cluster. In each successive iteration, a cluster is split into smaller clusters according to some measure until a termination condition is satisfied. Typically hierarchical approaches use the discriminative model of clustering, although generative interpretations of such models are possible.

Earlier hierarchical clustering methods such as AGNES and DIANA [KR90] suffered from the use of over-simplified measures for splitting and merging. Also, the merge or split operation was done irreversibly. This simple rigid approach resulted in erroneous clusters being found. In order to enhance the effectiveness of hierarchical clustering algorithms, recent methods have adopted one of the two following approaches. The first approach, represented by algorithms such as CURE [GRS98] and CHAMELEON [KHK99], utilizes a more complex principle when splitting or merging the clusters. Even though the split or merge operations are irreversible in these algorithms, they are very effective because of the use of somewhat involved techniques for the operations. The second approach, represented by algorithms such as BIRCH [ZRL96], is to obtain an initial result by using a hierarchical agglomerative algorithm and then refining the result using iterative relocation.

### 2.1.3 Density-based methods

Many partitioning methods cluster objects based on a variant of the Euclidean distance between objects from the cluster representatives. Such methods can find only "spherical" shaped clusters and encounter difficulty in discovering clusters of arbitrary shapes. In order to discover clusters of arbitrary shapes, some clustering methods have been developed based on the notion of *density*. These typically regard clusters as dense regions of objects in the data space that are separated by regions

of low density. Density-based methods can be used to filter out noise and discover clusters of arbitrary shape.

The DBSCAN algorithm [EKSX96] judges the density around the neighborhood of an object to be sufficiently high if the number of points within a distance $\epsilon$ of an object is greater than *MinPts* number of points. As the clusters discovered are dependent on the parameters $\epsilon$ and *MinPts*, DBSCAN relies on the user's ability to select a good set of parameters. To help overcome this problem, a cluster ordering method called OPTICS [ABKS99] was proposed. Rather than producing a data set clustering explicitly, OPTICS computes an augmented *cluster ordering* for automatic and interactive cluster analysis. Both these algorithms rely on spatial index structures and are not very efficient for high-dimensional data.

To handle higher dimensional data, DENCLUE [HK98] models the overall density of a point analytically as the sum of the influence functions of data points around it. To compute the sum of influence functions efficiently, a grid structure is utilized. Though DENCLUE seems to perform experimentally better than DBSCAN, a careful selection of clustering parameters is required.

As mentioned earlier, density-based methods like DBSCAN and OPTICS are index-based methods that face a breakdown in efficiency when the number of dimensions is high. To enhance the efficiency of clustering, a grid-based clustering approach uses a grid data structure. It quantizes the data space into a finite number of cells that form a grid structure on which all of the clustering are performed. The main advantage of the approach is its fast processing time that is typically independent of the number of data objects, yet dependent only on the number of cells in each dimension in the quantized space.

Some typical examples of the grid-based approach include STING [WYM97], that explores statistical information stored in the grid cells; WaveCluster [SCZ98], that clusters objects using a wavelet transform method; and CLIQUE [AGGR98],

that represents a grid- and density-based approach in high-dimensional data space. A grid-based approach is usually more efficient than a density-based approach especially in high-dimensions. On the other hand, the use of summarized information causes it to lose effectiveness under certain circumstances.

### 2.1.4 Graph-based methods

Graph-based methods transform the clustering problem into a combinatorial optimization problem that is solved using graph algorithms and related heuristics. Typically, if $n$ points are to be clustered, an undirected weighted graph $G = (V, E)$ is constructed whose vertices are the $n$ points so that $|V| = n$. An edge connecting any two vertices has a weight equal to a suitable similarity measure between the corresponding data-points. The choice of the similarity measure quite often depends on the problem domain, e.g., Jaccard coefficient for market baskets, normalized dot products for text, etc. With this setup, the clustering problem becomes one of finding a min-cut in the graph $G$. However, for $k > 2$, the min-cut problem is NP-complete [GJ79] and so most of the graph-based techniques are approximate solutions or good heuristics.

Some of the well known graph-based clustering algorithms include ROCK [GRS99], Chameleon [KHK99], Metis [KK98] and Opossum [SG00]. ROCK is an agglomerative hierarchical clustering technique for categorical attributes. It uses the binary Jaccard coefficient and a thresholding criterion to form unweighted edges connecting the data points. A key idea in ROCK is to define transitive neighbor relationship, i.e., in addition to using simple neighbors (according to the adjacency matrix $A$), all pairs having common neighbors (adjacency according to the matrix $A^T A$) are also considered neighbors. The common neighbors are used to define interconnectivity between clusters which is used to merge clusters. Chameleon starts with partitioning the data into a large number of small clusters by parti-

tioning the $v$-nearest neighbor graph. In the subsequent stages clusters are merged based on inter-connectivity. Metis is a fast multi-level framework for partitioning of weighted undirected graphs that has three stages: (a) coarsening of the graph by collapsing appropriately chosen vertices, (b) an initial partitioning of the coarsened graph, and (c) un-coarsening and refining the partitions of the graph. Opossum uses the extended Jaccard coefficient to form undirected weighted edges connecting the data-points and also allows weighing the data-points themselves. It performs sample-balanced or value-balanced clustering of the data using Metis to partition the weighted similarity graph.

## 2.2 Scalable Clustering

The past few years have witnessed a growing interest in clustering algorithms that are suitable for data-mining problems [JMF99, HKT01, Fas99, Gho03]. Clustering algorithms for data-mining problems must be extremely scalable. In addition, several data mining applications demand that the clusters obtained be balanced, i.e., be of approximately the same size or importance. There are several notable approaches that address the scalability issue. Some approaches try to build the clusters dynamically by maintaining sufficient statistics and other summarized information in main memory while minimizing the number of database scans involved. For example, Bradley *et al.* [BFR98b, BFR98a] propose out-of-core methods that scan the database once to form a summarized model (for instance, the size, sum and sum-squared values of potential clusters, and well as a small number of unallocated data-points) in main memory. Subsequent refinement based on this summarized information is then restricted to main memory operations without resorting to further disk scans. Another method with a similar flavor [ZRL96] compresses the data objects into many small subclusters using modified index trees and performs clustering with these subclusters. A different approach is to subsample the original data

before applying the actual clustering algorithms [CKPT92, GRS98]. Ways of effectively sampling large datasets have also been proposed [PF99]. [DH01] suggest using less number of points in each step of an iterative relocation optimization algorithm like `kmeans` as long as the model produced does not differ significantly from the one that would be obtained with full data.

Several of these methods are linear in the number of data-points $N$ as well as the number of clusters $k$ and hence scale very well. However their "sequential cluster building" nature prevents a global view of the emergent clustering that is needed for obtaining well-formed as well as reasonably balanced clusters. Even the venerable `KMeans` does not have any explicit way to guarantee that there is at least a certain minimum number of points per cluster, though it has been shown that `KMeans` has an implicit way of preventing highly skewed clusters [KMN97]. Experiments with `KMeans` show that it quite often generates some clusters that are empty or extremely small, specially when the data is in high dimensional ($> 100$) space. This is undesirable in certain practical scenarios where clusters should be of comparable sizes to be actually useful and actionable. For example, a direct marketing campaign often starts with segmenting customers into groups of roughly equal size or equal estimated revenue generation, (based on, say, market basket analysis, or purchasing behavior at a web site), so that the same number of sales teams (or marketing dollars) can be allocated to each segment. In large retail chains, one often desires product categories/groupings of comparable importance, since subsequent procedures such as shelf space allocation and product placement are influenced by the objective of allocating resources proportional to revenue or gross margins associated with the product groups. In clustering of a large corpus of documents to generate topic hierarchies, balancing greatly facilitates navigation by avoiding the generation of hierarchies that are highly skewed, with uneven depth in different parts of the hierarchy "tree" or having widely varying number of documents

at the leaf nodes.

There are a few existing approaches for obtaining balanced clusters. First, an agglomerative clustering method can be readily adapted so that once a cluster reaches a certain size in the bottom-up agglomeration process, it can be removed from further consideration. However, this may significantly impact cluster quality. Moreover, agglomerative clustering methods have a complexity of $\Omega(N^2)$ and hence does not scale well. A recent approach to obtain balanced clusters is to convert the clustering problem into a graph partitioning problem[KK98, SG00]. A similarity graph between data points is formed and partitioned by efficient algorithms such as METIS [KK98] that also incorporate a soft balancing constraint. Though this approach gives very good results, $\Omega(N^2)$ computation is required just to compute the similarity matrix. Another approach is to iterate over `kmeans` but do the cluster assignment by solving a minimum cost flow problem satisfying constraints [BBD00] on the cluster sizes. This approach is $O(N^3)$ and has even poorer scaling properties. A detailed study of constrained clustering in large databases is presented by Tung et. al. [TNLH01]. In particular, the authors show that the decision version of the balanced clustering problem is NP-hard, as expected, and go on to propose algorithms for getting to a locally optimal solution to the problem.

An alternative approach to obtain balanced clustering is via frequency sensitive competitive learning methods for clustering [AKCM90], where clusters of larger sizes are penalized so that points are less likely to get assigned to them. Such a scheme can be applied both in the batch as well as in the online settings [BG04]. Although frequency sensitive assignments can give fairly balanced clusters in practice, there is no obvious way to guarantee that every cluster will have at least a pre-specified number of points.

## 2.3 Learning with Bregman divergences

Bregman divergences [Bre67, CZ98] are a large class of distortion functions derived from convex functions that play a central role in this thesis. Given any differentiable, strictly convex function $\phi$, the Bregman divergence between $\mathbf{x}$ and $\mathbf{y}$, the domain of $\phi$, is defined as:

$$d_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - \langle \mathbf{x} - \mathbf{y}, \nabla\phi(\mathbf{y}) \rangle \ , \tag{2.5}$$

where $\nabla\phi(\mathbf{y})$ is the gradient of $\phi$ at $\mathbf{y}$. Bregman divergences have gained popularity in the machine learning literature over the past few years since they allow the unified analyses of a large class of algorithms while bringing out the key (convexity) properties that are used by such algorithms.

One important, and by now standard, usage of Bregman divergences in machine learning is for the analysis of regret bounds for online learning [GW00, KW01, HW01, AW01, FW00]. The key idea is as follows: Given a set of experts for a certain online decision problem, after going over $T$ cycles of decision making, each expert would have acquired a certain amount of gain (or loss) due to the particular decisions they made. For example, if the decision problem is that of predicting whether the weather will be "sunny" or "rainy" on a given day, and gain is measured simply by number of correct predictions over a period of time, each expert on weather prediction would have achieved a gain depending on its predictions. An algorithm that is trying to compete with the set of experts has to be as good as the best expert in the pool, although the best expert may change from day to day. The difference in performance measured by a certain loss (or distortion) function between the best expert and the algorithm is the regret of the algorithm. When the distortion is measured by a Bregman divergence, algorithms and corresponding analysis have been developed for getting bounds on the regret of the algorithm. Under very general

conditions, one can come up with algorithms with no regret on a per trial basis.

Another significant development using Bregman divergences was the unification of boosting and logistic regression and the discovery of a large class of parametric algorithms of which the previously known ones were special cases [CSS00]. The analyses also gave the first unconditional provable guarantees of the widely used boosting algorithms. Along with the close connections between online learning and boosting [FS97], the results of [CSS00] have given a strong foundation for and deep understanding of this class of learning algorithms.

A large class of statistical parametric regression models, called generalized linear models (GLMs) [MN89], that were earlier understood as log-likelihoods of parametric probability distributions, have been recently alternatively understood in terms of Bregman divergences [KW01]. Taking advantage of this viewpoint, one of the most popular dimensionality reduction techniques, called principal component analysis (PCA), has been successfully generalized to Bregman divergences [CDS01].

## 2.4 Evaluation of Clustering

There are two fundamentally different ways of evaluating the quality of results of clustering [Gho03]. Internal quality measures evaluate a clustering based on a function of the data and the clustering only. Typically, the quality is measured by the compactness of clusters or the intra-cluster similarities. In some cases, the separation between different clusters or the inter-cluster distances are measured. External quality measures evaluate a clustering based on external input, that was not used during by the clustering algorithm. If there are classification labels available for the data, one evaluates the agreement of the clustering with the class labels using various measures.

Almost all unsupervised clustering algorithms have an internal quality measure, i.e., one that does not depend on supervision such as labels, feedback in terms of

rewards etc. Examples of internal measures include average squared loss from a cluster representative [JD88], log-likelihood of a parametric probabilistic model [Bli98], some variant of a normalized edge-cut value of a pairwise similarity graph [KHK99], category utility function [Fis87] etc. A clustering algorithm typically tries to optimize its internal quality measure. Further, there are several unsupervised methods for comparing clusterings, e.g., Jaccard index, Rand index, Fowlkes-Mallows index, Mirkin metric, variation of information etc., exist in the literature [JD88, Mei03].

Since the predictive ability of clustering algorithms is often key to their successful application, external prediction-related quality measures are often appropriate. Several supervised measures such as purity, entropy, normalized mutual information, supervised F-measure etc. have been used in the literature (see [Gho03] for details). The main goal of these measures is to give an idea of the prediction ability of a clustering algorithm. It is however not clear how these measures are related to the error-rate of the clustering algorithm when used for prediction. Furthermore, none of these supervised measures help with cluster number selection, which is often a big issue for these supervised measures.

An information theoretic external validity measure motivated by the *minimum description length* (MDL) principle has been recently proposed [Dom01]. This measure correctly captures the predictive ability of a clustering algorithm on a train-set from the lossless compression and MDL viewpoint. Starting from a given clustering, the measure computes the number of extra bits required to exactly get the class label information. In spite of having several desirable properties, this measure has a few drawbacks: (i) the measure is not normalized to the interval [0,1] (and not easily normalized to exercise that interval) which is desirable in several settings, and many other quality measures actually satisfy this; and, (ii) the measure does not provide guarantees of prediction ability for test data.

## 2.5 Rate Distortion Theory

Rate distortion theory [Ber71, BG98] deals with the fundamental limits of quantizing a stochastic source $X \sim p(x),\ x \in \mathcal{X}$, using a random variable $\hat{X}$ over a reproduction alphabet $\hat{\mathcal{X}}$ typically assumed to embed the source alphabet $\mathcal{X}$, i.e., $\mathcal{X} \subseteq \hat{\mathcal{X}}$. In the rate distortion setting, the performance of a quantization scheme is determined in terms of the rate, i.e., the average number of bits for encoding a symbol, and the expected distortion between the source and the reproduction random variables based on an appropriate distortion function $d : \mathcal{X} \times \hat{\mathcal{X}} \mapsto \mathbb{R}_+$. The central problem in rate distortion theory [CT91] is to compute the rate distortion function $R(D)$, which is defined as the minimum achievable rate for a specified level of expected distortion $D$, and can be mathematically expressed as

$$R(D) = \min_{p(\hat{x}|x):E_{X,\hat{X}}[d(X,\hat{X})]\leq D} I(X;\hat{X}) \ , \tag{2.6}$$

where $I(X;\hat{X})$ is the mutual information of $X$ and $\hat{X}$.

The rate distortion problem is a convex problem that involves optimizing over the probabilistic assignments $p(\hat{x}|x)$ and can be theoretically solved using the celebrated Blahut-Arimoto algorithm [Ari72, Bla72, Csi74, CT91]. However, numerical computation of the rate distortion function through the Blahut-Arimoto algorithm is often infeasible in practice, primarily due to the lack of knowledge of the optimal support of the reproduction random variable $\hat{X}$. Analytic closed form expressions of the rate distortion function exist only for certain well-behaved source and distortion measure combinations. Therefore, exact computation of the rate distortion function has remained a difficult problem.

## 2.6 Stochastic Prediction

The problem of predicting a random variable based on available information arises in many contexts. Let $X$ denote the random variable to be predicted and let $Z$ be the information one has about $X$ from observations. For example, $X$ may denote all the relevant information about a flying aeroplane, and $Z$ denotes the information a pilot has from the readings in the cockpit [Kni94, section 2.1]. Now, given the observation $Z$, what is the best guess about $X$ one can make?

To put the problem into a mathematical framework, let $(\Omega, \mathcal{F}, \mathbf{P})$ be a probability space and let $X$ be an $\mathcal{F}$-measurable random variable that one wishes to predict. If $Z$ is the observation random variable, the available partial information about $X$ that can be obtained by observing $Z$ is represented by $\sigma(Z)$. Mathematically, $\sigma(Z)$ is the $\sigma$-algebra generated by $Z$ and contains all Borel-measurable functions of $Z$. Now, since $Z$ is the observation, any predictor $Y$ of $X$ has to be a function of $Z$, so that $Y \in \sigma(Z)$. A basic question in stochastic prediction is: among all functions of $Z$, which one is the best predictor of $X$?

The notion of *best* is usually specified by a non-negative loss function $F$ and achieved by solving a corresponding minimization problem. More precisely, the best predictor of $X$ is defined as the minimizer of $E[F(X,Y)]$ over all $Y \in \sigma(Z)$. A particularly important case is when $F$ is the so called $\mathbb{L}^2$-loss function, also known as the squared error, i.e., $F(x,y) \doteq \|x-y\|^2$. It is well known [KT74, Kni94, Wil91] that the corresponding *unique* best predictor is given by the conditional expectation. In other words,

$$\operatorname*{argmin}_{Y \in \sigma(Z)} E\left[\|X-Y\|^2\right] = E[X|Z] \ .$$

This makes conditional expectation crucially important for prediction.

# Chapter 3

# Hard Clustering with Bregman Divergences

Several algorithms for solving particular versions of parametric clustering problems have been developed over the years. As far as hard clustering algorithms are concerned, the most well-known is the iterative relocation scheme for the Euclidean `kmeans` algorithm [Mac67, JD88, DHS00]. Another widely used clustering algorithm with a similar scheme is the Linde-Buzo-Gray (LBG) algorithm [LBG80, BGGM80] based on the Itakura-Saito distance, which has been used in the signal-processing community for clustering speech data. The recently proposed information theoretic clustering algorithm [DMK03] for clustering probability distributions also has a similar flavor.

For certain distortion functions, e.g., squared Euclidean distance, KL divergence [DMK03], Itakura-Saito distance [BGGM80] etc., the clustering problem can be solved using appropriate `kmeans` type iterative relocation schemes. In this context, an interesting question to ask is: *what class of distortion functions admit such an iterative relocation scheme where a global objective function based on the distor-*

[0]The work presented in this chapter has earlier appeared in part as [BMDG04].

*tion with respect to cluster centroids*[1] *is progressively decreased?* In this chapter, we provide an answer to this question: we show that *such a scheme works for arbitrary Bregman divergences*. The scope of this result is vast since Bregman divergences include a large number of useful loss functions such as squared loss, KL-divergence, logistic loss, Mahalanobis distance, Itakura-Saito distance, I-divergence, etc.

In this chapter, we pose the hard clustering problem as one of obtaining an optimal quantization in terms of minimizing the loss in *Bregman information*, a quantity motivated by rate distortion theory. A simple analysis then yields a version of the loss function that readily suggests a natural algorithm to solve the clustering problem for arbitrary Bregman divergences. Partitional hard clustering to minimize the loss in *mutual information*, otherwise known as information theoretic clustering [DMK03], is seen to be a special case of our approach. Thus, this chapter unifies several parametric partitional clustering approaches.

## 3.1 Preliminaries

In this section, we define the Bregman divergence corresponding to a strictly convex function and present some examples.

**Definition 1** [Bre67, CZ98] Let $\phi : \mathcal{S} \mapsto \mathbb{R}$, $\mathcal{S} = \mathrm{dom}(\phi)$, be a strictly convex function defined on a convex set $\mathcal{S} \subseteq \mathbb{R}^d$ such that $\phi$ is differentiable on $\mathrm{ri}(\mathcal{S})$, assumed to be nonempty. The *Bregman divergence* $d_\phi : \mathcal{S} \times \mathrm{ri}(\mathcal{S}) \mapsto [0, \infty)$ is defined as

$$d_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - \langle \mathbf{x} - \mathbf{y}, \nabla\phi(\mathbf{y}) \rangle \ ,$$

where $\nabla\phi(\mathbf{y})$ represents the gradient vector of $\phi$ evaluated at $\mathbf{y}$.

**Example 1** Squared Euclidean distance is perhaps the simplest and most widely

[1] We use the term "cluster centroid" to denote the expectation of the data points in that cluster.

used Bregman divergence. The underlying function $\phi(\mathbf{x}) = \langle \mathbf{x}, \mathbf{x} \rangle$ is strictly convex, differentiable on $\mathbb{R}^d$ and

$$\begin{aligned} d_\phi(\mathbf{x}, \mathbf{y}) &= \langle \mathbf{x}, \mathbf{x} \rangle - \langle \mathbf{y}, \mathbf{y} \rangle - \langle \mathbf{x} - \mathbf{y}, \nabla\phi(\mathbf{y}) \rangle \\ &= \langle \mathbf{x}, \mathbf{x} \rangle - \langle \mathbf{y}, \mathbf{y} \rangle - \langle \mathbf{x} - \mathbf{y}, 2\mathbf{y} \rangle \\ &= \langle \mathbf{x} - \mathbf{y}, \mathbf{x} - \mathbf{y} \rangle = \|\mathbf{x} - \mathbf{y}\|^2. \end{aligned}$$

**Example 2** Another widely used Bregman divergence is the KL-divergence. If $\mathbf{p}$ is a discrete probability distribution so that $\sum_{j=1}^d p_j = 1$, the negative entropy $\phi(\mathbf{p}) = \sum_{j=1}^d p_j \log_2 p_j$ is a convex function. The corresponding Bregman divergence is

$$\begin{aligned} d_\phi(\mathbf{p}, \mathbf{q}) &= \sum_{j=1}^d p_j \log_2 p_j - \sum_{j=1}^d q_j \log_2 q_j - \langle \mathbf{p} - \mathbf{q}, \nabla\phi(\mathbf{q}) \rangle \\ &= \sum_{j=1}^d p_j \log_2 p_j - \sum_{j=1}^d q_j \log_2 q_j - \sum_{j=1}^d (p_j - q_j)(\log_2 q_j + \log_2 e) \\ &= \sum_{j=1}^d p_j \log_2 \left( \frac{p_j}{q_j} \right) - \log_2 e \sum_{j=1}^d (p_j - q_j) \\ &= KL(\mathbf{p} \| \mathbf{q}) \ , \end{aligned}$$

the KL-divergence between the two distributions as $\sum_{j=1}^d q_j = \sum_{j=1}^d p_j = 1$.

**Example 3** Itakura-Saito distance is another Bregman divergence that is widely used in signal processing. If $F(e^{j\theta})$ is the power spectrum[2] of a signal $f(t)$, then the functional $\phi(F) = -\frac{1}{2\pi} \int_{-\pi}^{\pi} \log(F(e^{j\theta})) d\theta$ is convex in $F$ and corresponds to the negative entropy rate of the signal assuming it was generated by a stationary Gaussian process [Pal97, CT91]. The Bregman divergence between $F(e^{j\theta})$ and $G(e^{j\theta})$

[2]Note that $F(\cdot)$ is a function and it is possible to extend the notion of Bregman divergences to the space of functions [Csi95, GD04].

Table 3.1: Bregman divergences generated from some convex functions.

| **Domain** | $\phi(\mathbf{x})$ | $d_\phi(\mathbf{x},\mathbf{y})$ | **Divergence** |
|---|---|---|---|
| $\mathbb{R}$ | $x^2$ | $(x-y)^2$ | Square loss |
| $\mathbb{R}_+$ | $x\log x$ | $x\log(\frac{x}{y})-(x-y)$ | |
| $[0,1]$ | $x\log x+(1-x)\log(1-x)$ | $x\log(\frac{x}{y})+(1-x)\log(\frac{1-x}{1-y})$ | Logistic loss [3] |
| $\mathbb{R}_{++}$ | $-\log x$ | $\frac{x}{y}-\log(\frac{x}{y})-1$ | Itakura-Saito distance |
| $\mathbb{R}$ | $e^x$ | $e^x-e^y-(x-y)e^y$ | |
| $\mathbb{R}^d$ | $\|\mathbf{x}\|^2$ | $\|\mathbf{x}-\mathbf{y}\|^2$ | Squared Euclidean distance |
| $\mathbb{R}^d$ | $\mathbf{x}^TA\mathbf{x}$ | $(\mathbf{x}-\mathbf{y})^TA(\mathbf{x}-\mathbf{y})$ | Mahalanobis distance [4] |
| $d$-Simplex | $\sum_{j=1}^d x_j\log_2 x_j$ | $\sum_{j=1}^d x_j\log_2(\frac{x_j}{y_j})$ | KL-divergence |
| $\mathbb{R}_+^d$ | $\sum_{j=1}^d x_j\log x_j$ | $\sum_{j=1}^d x_j\log(\frac{x_j}{y_j})-\sum_{j=1}^d(x_j-y_j)$ | Generalized I-divergence |

(the power spectrum of another signal $g(t)$) is given by

$$\begin{aligned} d_\phi(F,G) &= \frac{1}{2\pi}\int_{-\pi}^{\pi}\Big(-\log(F(e^{j\theta}))+\log(G(e^{j\theta}))-(F(e^{j\theta})-G(e^{j\theta}))\nabla\phi(G)\Big)\,d\theta \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi}\left(-\log\left(\frac{F(e^{j\theta})}{G(e^{j\theta})}\right)+\frac{F(e^{j\theta})}{G(e^{j\theta})}-1\right)d\theta, \end{aligned}$$

which is exactly the Itakura-Saito distance between the power spectra $F(e^{j\theta})$ and $G(e^{j\theta})$ and can also be interpreted as the I-divergence [Csi91] between the generating processes under the assumption that they are equal mean, stationary Gaussian processes [KK80].

Table 3.1 contains a list of some common convex functions and their corresponding Bregman divergences. Bregman divergences have several interesting and useful properties, such as non-negativity, convexity in the first argument, etc. For details see Appendix A.

In the next section, we introduce a new concept called the Bregman information of a random variable based on ideas from Shannon's rate distortion theory.

[3] For $x\in\{0,1\}$ (Bernoulli) and $y\in(0,1)$ (posterior probability for $x=1$), we have $x\log(\frac{x}{y})+(1-x)\log(\frac{1-x}{1-y})=\log(1+\exp(-f(x)g(y)))$, i.e., the logistic loss with $f(x)=2x-1$ and $g(y)=\log(\frac{y}{1-y})$.

[4] The matrix $A$ is assumed to be a positive definite matrix; $(\mathbf{x}-\mathbf{y})^TA(\mathbf{x}-\mathbf{y})$ is called the Mahalanobis distance when $A$ is the inverse of the covariance matrix.

Then we motivate the Bregman hard clustering problem as a quantization problem that involves minimizing the loss in Bregman information and show its equivalence to a more direct formulation, i.e., the problem of finding a partitioning and a representative for each of the partitions such that the expected Bregman divergence of the data points from their representatives is minimized. We then present a clustering algorithm that generalizes the iterative relocation scheme of `KMeans` to monotonically decrease the loss in Bregman information.

## 3.2 Bregman Information

The dual formulation of Shannon's celebrated rate distortion problem [CT91, GV03] involves finding a coding scheme with a given rate, i.e., average number of bits per symbol, such that the expected distortion between the source random variable and the decoded random variable is minimized. The achieved distortion is called the *distortion rate function*, which is the infimum distortion achievable for a given rate. Now consider a random variable $X$ that takes values in a finite set $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{S} \subseteq \mathbb{R}^d$ ($\mathcal{S}$ is convex) following a discrete probability measure $\nu$. Let the distortion be measured by a Bregman divergence $d_\phi$. Consider a simple encoding scheme that represents the random variable by a constant vector $\mathbf{s}$, i.e., codebook size is one, or rate is zero. The solution to the rate-distortion problem in this case is the trivial assignment. The corresponding distortion-rate function is given by $E_\nu[d_\phi(X, \mathbf{s})]$ that depends on the choice of the representative $\mathbf{s}$ and can be optimized by picking the right representative. We call this optimal distortion-rate function the *Bregman information* of the random variable $X$ for the Bregman divergence $d_\phi$ and denote it by $I_\phi(X)$, i.e.,

$$I_\phi(X) = \min_{\mathbf{s} \in \mathrm{ri}(\mathcal{S})} E_\nu[d_\phi(X, \mathbf{s})] = \min_{\mathbf{s} \in \mathrm{ri}(\mathcal{S})} \sum_{i=1}^{n} \nu_i \; d_\phi(\mathbf{x}_i, \mathbf{s}) \; . \tag{3.1}$$

The optimal vector $\mathbf{s}$ that achieves the minimal distortion will be called the *Bregman representative* or, simply the *representative* of $X$. The following theorem states that this representative always exists, is uniquely determined and, surprisingly, *does not depend* on the choice of Bregman divergence. In fact, the minimizer is just the expectation of the random variable $X$.

**Proposition 1** *Let $X$ be a random variable that take values in $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{S} \subseteq \mathbb{R}^d$ following a probability measure $\nu$ such that $E_\nu[X] \in \mathrm{ri}(\mathcal{S})$. Given a Bregman divergence $d_\phi : \mathcal{S} \times \mathrm{ri}(\mathcal{S}) \mapsto [0, \infty)$, the problem*

$$\min_{\mathbf{s} \in \mathrm{ri}(\mathcal{S})} E_\nu[d_\phi(X, \mathbf{s})] \tag{3.2}$$

*has a unique minimizer given by $\mathbf{s}^\dagger = \boldsymbol{\mu} = E_\nu[X]$.*

*Proof:* The objective function we are trying to minimize is $J_\phi(\mathbf{s}) = E_\nu[d_\phi(X, \mathbf{s})] = \sum_{i=1}^n \nu_i d_\phi(\mathbf{x}_i, \mathbf{s})$. Since $\boldsymbol{\mu} = E_\nu[X] \in \mathrm{ri}(S)$, the objective function is well-defined at $\boldsymbol{\mu}$. Now, $\forall \mathbf{s} \in \mathrm{ri}(\mathcal{S})$,

$$\begin{aligned}
J_\phi(\mathbf{s}) - J_\phi(\boldsymbol{\mu}) &= \sum_{i=1}^n \nu_i d_\phi(\mathbf{x}_i, \mathbf{s}) - \sum_{i=1}^n \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}) \\
&= \phi(\boldsymbol{\mu}) - \phi(\mathbf{s}) - \left\langle \sum_{i=1}^n \nu_i \mathbf{x}_i - \mathbf{s}, \nabla\phi(\mathbf{s}) \right\rangle + \left\langle \sum_{i=1}^n \nu_i \mathbf{x}_i - \boldsymbol{\mu}, \nabla\phi(\boldsymbol{\mu}) \right\rangle \\
&= \phi(\boldsymbol{\mu}) - \phi(\mathbf{s}) - \langle \boldsymbol{\mu} - \mathbf{s}, \nabla\phi(\mathbf{s}) \rangle \\
&= d_\phi(\boldsymbol{\mu}, \mathbf{s}) \geq 0,
\end{aligned}$$

with equality only when $\mathbf{s} = \boldsymbol{\mu}$ by the strict convexity of $\phi$ (Appendix A, Property 1). Hence, $\boldsymbol{\mu}$ is the unique minimizer of $J_\phi$. ∎

Note that the minimization in (3.2) is with respect to the second argument of $d_\phi$. Proposition 1 is somewhat surprising since Bregman divergences are not necessarily convex in the second argument as the following example demonstrates.

**Example 4** Consider $\phi(\mathbf{x}) = \sum_{j=1}^{3} x_j^3$ defined on $\mathbb{R}_+^3$ so that $d_\phi(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{3}(x_j^3 - y_j^3 - 3(x_j - y_j)y_j^2)$. For the random variable $X$ distributed uniformly over the set $\mathcal{X} = \{(1,1,1),(2,2,2),(3,3,3),(4,4,4),(5,5,5)\}$,

$$J_\phi(\mathbf{y}) = 135 + 2\sum_{j=1}^{3} y_j^3 - 9\sum_{j=1}^{3} y_j^2 \ ,$$

which is clearly not convex in $\mathbf{y}$ since the Hessian $\nabla^2 J_\phi(\mathbf{y}) = \mathrm{diag}(12\mathbf{y} - 18)$, is not positive definite. However, $J_\phi(\mathbf{y})$ is uniquely minimized by $\mathbf{y} = (3,3,3)$, i.e., the expectation of the random variable $X$.

Interestingly, the converse of Proposition 1 is also true, i.e., for all random variables $X$, if $E[X]$ minimizes the expected distortion of $X$ to a fixed constant for a smooth distortion function $F(x,y)$ (see Chapter 9 for details), then $F(x,y)$ has to be a Bregman divergence [BGW05]. Thus, Bregman divergences are *exhaustive* with respect to the property proved in Proposition 1.

Using Proposition 1, we can now give a more direct definition of Bregman information as follows:

**Definition 2** Let $X$ be a random variable that takes values in $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{S}$ following a probability measure $\nu$. Let $\boldsymbol{\mu} = E_\nu[X] = \sum_{i=1}^n \nu_i \mathbf{x}_i \in \mathrm{ri}(\mathcal{S})$ and let $d_\phi : \mathcal{S} \times \mathrm{ri}(\mathcal{S}) \mapsto [0, \infty)$ be a Bregman divergence. Then the **Bregman Information** of $X$ in terms of $d_\phi$ is defined as

$$I_\phi(X) = E_\nu[d_\phi(X, \boldsymbol{\mu})] = \sum_{i=1}^{n} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}) \ .$$

**Example 5** (**Variance**) Let $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ be a set in $\mathbb{R}^d$, and consider the uniform measure, i.e., $\nu_i = \frac{1}{n}$, over $\mathcal{X}$. The Bregman information of $X$ with squared

Euclidean distance as the Bregman divergence is given by

$$I_\phi(X) = \sum_{i=1}^{n} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}) = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{x}_i - \boldsymbol{\mu}\|^2,$$

which is just the sample variance.

**Example 6** (**Mutual Information**) This example shows that mutual information is a special case of Bregman divergence. By definition, the mutual information $I(U;V)$ between two random variables $U$ and $V$ with joint distribution $\{\{p(u_i, v_j)\}_{i=1}^n\}_{j=1}^m$ is given by

$$\begin{aligned} I(U;V) &= \sum_{i=1}^{n}\sum_{j=1}^{m} p(u_i, v_j) \log \frac{p(u_i, v_j)}{p(u_i)p(v_j)} \\ &= \sum_{i=1}^{n} p(u_i) \sum_{j=1}^{m} p(v_j|u_i) \log \frac{p(v_j|u_i)}{p(v_j)} \\ &= \sum_{i=1}^{n} p(u_i) KL(\ p(V|u_i)\ \|\ p(V)\ )\ . \end{aligned}$$

Consider a random variable $Z_u$ that takes values in the set of probability distributions $\mathcal{Z}_u = \{p(V|u_i)\}_{i=1}^n$ following the probability measure $\{\nu_i\}_{i=1}^n = \{p(u_i)\}_{i=1}^n$ over this set. The mean (distribution) of $Z_u$ is given by

$$\boldsymbol{\mu} = E_\nu[p(V|u)] = \sum_{i=1}^{n} p(u_i)p(V|u_i) = \sum_{i=1}^{n} p(u_i, V) = p(V)\ .$$

Hence,

$$\begin{aligned} I(U;V) &= \sum_{i=1}^{n} p(u_i) KL(\ p(V|u_i)\ \|\ p(V)\ ) \\ &= \sum_{i=1}^{n} \nu_i d_\phi(p(V|u_i), \boldsymbol{\mu}) = I_\phi(Z_u), \end{aligned}$$

i.e., mutual information is the Bregman information of $Z_u$ when $d_\phi$ is the KL-divergence. Further, for a random variable $Z_v$ that takes values in the set of probability distributions $\mathcal{Z}_v = \{p(U|v_j)\}_{j=1}^m$ following the probability measure $\{\nu_j\}_{j=1}^m = \{p(v_j)\}_{j=1}^m$ over this set, one can similarly show that $I(U;V) = I_\phi(Z_v)$. The Bregman information of $Z_u$ and $Z_v$ can also be interpreted as the Jensen-Shannon divergence of the sets $\mathcal{Z}_u$ and $\mathcal{Z}_v$ [DMK03].

**Example 7** The Bregman information corresponding to the Itakura-Saito distance also has a useful interpretation. Let $\mathcal{F} = \{F_i\}_{i=1}^n$ be a set of power spectra corresponding to $n$ different signals, and let $\nu$ be a probability measure on $\mathcal{F}$. Then the Bregman information of a random variable $F$ that takes values in $\mathcal{F}$ following $\nu$, with Itakura-Saito distance as the Bregman divergence, is given by

$$\begin{aligned} I_\phi(F) &= \sum_{i=1}^n \nu_i d_\phi(F_i, \bar{F}) = \sum_{i=1}^n \frac{\nu_i}{2\pi} \int_{-\pi}^{\pi} \left( -\log\left(\frac{F_i(e^{j\theta})}{\bar{F}(e^{j\theta})}\right) + \frac{F_i(e^{j\theta})}{\bar{F}(e^{j\theta})} - 1 \right) d\theta \\ &= -\frac{1}{2\pi} \int_{-\pi}^{\pi} \sum_{i=1}^n \nu_i \log\left(\frac{F_i(e^{j\theta})}{\bar{F}(e^{j\theta})}\right) d\theta, \end{aligned}$$

where $\bar{F}$ is the marginal average power spectrum. Based on the connection between the corresponding convex function $\phi$ and the negative entropy of Gaussian processes [CT91, Pal97], it can be shown that the Bregman information $I_\phi(F)$ is the Jensen-Shannon divergence of the generating processes under the assumption that they are equal mean, stationary Gaussian processes. Further, consider a $n$-class signal classification problem where each class of signals is assumed to be generated by a certain Gaussian process. Now, if $P_e(t)$ is the optimal Bayes error for this classification problem averaged upto time $t$, then $P_e(t)$ is bounded above and below by functions of the Chernoff coefficient $B(t)$ [KK80] of the generating Gaussian processes. The asymptotic value of this Chernoff coefficient as $t$ tends to $\infty$ is a

function of the Bregman information of $F$, i.e.,

$$\lim_{t\to\infty} B(t) = \exp(-\frac{1}{2} I_\phi(F)).$$

and is directly proportional to the optimal Bayes error.

**Jensen's Inequality and Bregman Information**

An alternative interpretation of Bregman information can also be made in terms of Jensen's inequality [CT91]. Given any convex function $\phi$, for any random variable $X$, Jensen's inequality states that

$$E[\phi(X)] \geq \phi(E[X]) \ .$$

A direct calculation using the definition of Bregman information shows that [BGW04]

$$\begin{aligned} E[\phi(X)] - \phi(E[X]) &\overset{(a)}{=} E[\phi(X)] - \phi(E[X]) - E[\langle X - E[X], \nabla\phi(E[X])\rangle] \\ &\overset{(b)}{=} E[\phi(X) - \phi(E[X]) - \langle X - E[X], \nabla\phi(E[X])\rangle] \\ &= E[d_\phi(X, E[X])] \ = \ I_\phi(X) \ \geq \ 0 \ , \end{aligned}$$

where (a) follows since the last term is 0, and (b) follows by the linearity of expectation. Thus, the difference between the two sides of Jensen's inequality is exactly equal to the Bregman information.

### 3.2.1 Clustering Formulation

Let $X$ be a random variable that takes values in $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ following the probability measure $\nu$. When $X$ has a large Bregman information, it may not suffice to encode $X$ using a single representative since a low quantization error may be desired. In such a situation, a natural goal is to split the set $\mathcal{X}$ into $k$ disjoint

partitions $\{\mathcal{X}_h\}_{h=1}^k$, each with its own Bregman representative, such that a random variable $M$ over the partition representatives serves as an appropriate quantization of $X$. Let $\mathcal{M} = \{\boldsymbol{\mu}_h\}_{h=1}^k$ denote the set of representatives, and $\pi = \{\pi_h\}_{h=1}^k$ with $\pi_h = \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i$ denote the induced probability measure on $\mathcal{M}$. Then the induced random variable $M$ takes values in $\mathcal{M}$ following $\pi$.

The quality of the quantization $M$ can be measured by the expected Bregman divergence between $X$ and $M$, i.e., $E_{X,M}[d_\phi(X, M)]$. Since $M$ is a deterministic function of $X$, the expectation is actually over the distribution of $X$, so that

$$E_X[d_\phi(X, M)] = \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h) = \sum_{h=1}^{k} \pi_h \sum_{\mathbf{x}_i \in \mathcal{X}_h} \frac{\nu_i}{\pi_h} d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h) = E_\pi[I_\phi(X_h)] \ ,$$

where $X_h$ is the random variable that takes values in the partition $\mathcal{X}_h$ following a probability distribution $\frac{\nu_i}{\pi_h}$, and $I_\phi(X_h)$ is the Bregman information of $X_h$. Thus, the quality of the quantization is equal to the expected Bregman information of the partitions.

An alternative way of measuring the quality of the quantization $M$ can be formulated from an infomation theoretic viewpoint. In information-theoretic clustering [DMK03], the quality of the partitioning is measured in terms of the loss in mutual information resulting from the quantization of the original random variable $X$. Extending this formulation, we can measure the quality of the quantization $M$ by the loss in Bregman information due to the quantization, i.e., by $I_\phi(X) - I_\phi(M)$. For $k = n$, the best choice is of course $M = X$ with no loss in Bregman information. For $k = 1$, the best quantization is to pick $E_\nu[X]$ with probability 1, incurring a loss of $I_\phi(X)$. For intermediate values of $k$, the solution is less obvious.

Interestingly the two possible formulations outlined above turns out to be identical (see Theorem 1 below). We choose the information theoretic viewpoint to pose the problem, since we will study the connections of both the hard and soft

clustering problems to rate distortion theory in Chapter 8. Thus we define the *Bregman hard clustering problem* as that of finding a partitioning of $\mathcal{X}$, or, equivalently, finding the random variable $M$, such that *the loss in Bregman information* due to quantization, $L_\phi(M) = I_\phi(X) - I_\phi(M)$, *is minimized*. Typically, clustering algorithms assume a uniform measure, i.e., $\nu_i = \frac{1}{n}, \forall i$, over the data, which is clearly a special case of our formulation.

The following theorem shows that the loss in Bregman information and the expected Bregman information of the partitions are equal.

**Theorem 1** *Let $X$ be a random variable that takes values in $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{S} \subseteq \mathbb{R}^d$ following the positive probability measure $\nu$. Let $\{\mathcal{X}_h\}_{h=1}^k$ be a partitioning of $\mathcal{X}$ and let $\pi_h = \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i$ be the induced measure $\pi$ on the partitions. Let $X_h$ be the random variable that takes values in $\mathcal{X}_h$ following $\frac{\nu_i}{\pi_h}$ for $\mathbf{x}_i \in \mathcal{X}_h$, for $h = 1, \ldots, k$. Let $\mathcal{M} = \{\boldsymbol{\mu}_h\}_{h=1}^k$ denote the set of representatives of $\{X_h\}_{h=1}^k$, and $M$ be a random variable that takes values in $\mathcal{M}$ following $\pi$. Then,*

$$L_\phi(M) \;=\; I_\phi(X) - I_\phi(M) \;=\; E_\pi[I_\phi(X_h)] \;=\; \sum_{h=1}^{k} \pi_h \sum_{\mathbf{x}_i \in \mathcal{X}_h} \frac{\nu_i}{\pi_h}\, d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h)\ .$$

*Proof:* By definition,

$$\begin{aligned}
I_\phi(X) \;&=\; \sum_{i=1}^{n} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}) \;=\; \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}) \\
&= \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i \left\{\phi(\mathbf{x}_i) - \phi(\boldsymbol{\mu}) - \langle \mathbf{x}_i - \boldsymbol{\mu}, \nabla\phi(\boldsymbol{\mu})\rangle\right\} \\
&= \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i \left\{\phi(\mathbf{x}_i) - \phi(\boldsymbol{\mu}_h) - \langle \mathbf{x}_i - \boldsymbol{\mu}_h, \nabla\phi(\boldsymbol{\mu}_h)\rangle + \langle \mathbf{x}_i - \boldsymbol{\mu}_h, \nabla\phi(\boldsymbol{\mu}_h)\rangle\right. \\
&\qquad \left. + \phi(\boldsymbol{\mu}_h) - \phi(\boldsymbol{\mu}) - \langle \mathbf{x}_i - \boldsymbol{\mu}_h + \boldsymbol{\mu}_h - \boldsymbol{\mu}, \nabla\phi(\boldsymbol{\mu})\rangle\right\}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i \{\ d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h) + d_\phi(\boldsymbol{\mu}_h, \boldsymbol{\mu}) + \langle \mathbf{x}_i - \boldsymbol{\mu}_h, \nabla\phi(\boldsymbol{\mu}_h) - \nabla\phi(\boldsymbol{\mu}) \rangle\ \} \\
&= \sum_{h=1}^{k} \pi_h \sum_{\mathbf{x}_i \in \mathcal{X}_h} \frac{\nu_i}{\pi_h} d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h) + \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i d_\phi(\boldsymbol{\mu}_h, \boldsymbol{\mu}) \\
&\qquad + \sum_{h=1}^{k} \pi_h \sum_{\mathbf{x}_i \in \mathcal{X}_h} \frac{\nu_i}{\pi_h} \langle \mathbf{x}_i - \boldsymbol{\mu}_h, \nabla\phi(\boldsymbol{\mu}_h) - \nabla\phi(\boldsymbol{\mu}) \rangle \\
&= \sum_{h=1}^{k} \pi_h I_\phi(X_h) + \sum_{h=1}^{k} \pi_h d_\phi(\boldsymbol{\mu}_h, \boldsymbol{\mu}) \\
&\qquad + \sum_{h=1}^{k} \pi_h \left\langle \sum_{\mathbf{x}_i \in \mathcal{X}_h} \frac{\nu_i}{\pi_h} \mathbf{x}_i - \boldsymbol{\mu}_h, \nabla\phi(\boldsymbol{\mu}_h) - \nabla\phi(\boldsymbol{\mu}) \right\rangle \\
&= E_\pi[I_\phi(X_h)] + I_\phi(M),
\end{aligned}
$$

since $\sum_{\mathbf{x}_i \in \mathcal{X}_h} \frac{\nu_i}{\pi_h} \mathbf{x}_i = \boldsymbol{\mu}_h$. ∎

Note that $I_\phi(X)$ can be interpreted as the "total Bregman information", and $I_\phi(M)$ can be interpreted as the "between-cluster Bregman information" since it is a measure of divergence between the cluster representatives, while $L_\phi(M)$ can be interpreted as the "within-cluster Bregman information". Thus Theorem 1 states that the total Bregman information equals the sum of the within-cluster Bregman information and between-cluster Bregman information. This is a generalization of the corresponding result for squared Euclidean distances [DHS00].

Using theorem 1, the Bregman clustering problem of minimizing the loss in Bregman information can be written as

$$
\min_M \ (I_\phi(X) - I_\phi(M)) = \min_M \ \left( \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h) \right) \ . \tag{3.3}
$$

Thus, the loss in Bregman information is minimized if the set of representatives $\mathcal{M}$ is such that the expected Bregman divergence of points in the original set $\mathcal{X}$ to their corresponding representatives is minimized. We shall investigate the relationship of

this formulation to rate distortion theory in Chapter 8.

### 3.2.2 Clustering Algorithm

The objective function given in (3.3) suggests a natural iterative relocation algorithm for solving the Bregman hard clustering problem (see Algorithm 1). It is easy to see that classical `kmeans`, the LBG algorithm [BGGM80] and the information theoretic clustering algorithm [DMK03] are special cases of Bregman hard clustering for squared Euclidean distance, Itakura-Saito distance and KL-divergence respectively. The following propositions prove the convergence of the Bregman hard clustering algorithm.

**Proposition 2** *The Bregman hard clustering algorithm (Algorithm 1) monotonically decreases the loss function in (3.3).*

*Proof:* Let $\{\mathcal{X}_h^{(t)}\}_{h=1}^k$ be the partitioning of $\mathcal{X}$ after the $t^{th}$ iteration and let $\mathcal{M}^{(t)} = \{\boldsymbol{\mu}_h^{(t)}\}_{h=1}^k$ be the corresponding set of cluster representatives. Then,

$$\begin{aligned} L_\phi(M^{(t)}) &= \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h^{(t)}} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h^{(t)}) \overset{(a)}{\geq} \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h^{(t)}} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_{h^\dagger(\mathbf{x}_i)}^{(t)}) \\ &\overset{(b)}{\geq} \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h^{(t+1)}} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h^{(t+1)}) = L_\phi(M^{(t+1)}), \end{aligned}$$

where (a) follows from the assignment step, and (b) follows from the re-estimation step and Proposition 1. Note that if equality holds, i.e., if the loss function value is equal at consecutive iterations, then the algorithm will terminate. ∎

**Proposition 3** *The Bregman hard clustering algorithm (Algorithm 1) terminates in a finite number of steps at a partition that is locally optimal, i.e., the total loss cannot be decreased by either (a) the assignment step or by (b) changing the means of any existing clusters.*

**Algorithm 1** Bregman Hard Clustering

**Input:** Set $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{S} \subseteq \mathbb{R}^d$, probability measure $\nu$ over $\mathcal{X}$, Bregman divergence $d_\phi : \mathcal{S} \times \text{int}(\mathcal{S}) \mapsto \mathbb{R}$, number of clusters $k$.

**Output:** $\mathcal{M}^\dagger$, local minimizer of $L_\phi(\mathcal{M}) = \sum_{h=1}^k \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h)$ where $\mathcal{M} = \{\boldsymbol{\mu}_h\}_{h=1}^k$, hard partitioning $\{\mathcal{X}_h\}_{h=1}^k$ of $\mathcal{X}$.

**Method:**

Initialize $\{\boldsymbol{\mu}_h\}_{h=1}^k$ with $\boldsymbol{\mu}_h \in \text{int}(\mathcal{S})$ (one possible initialization is to choose $\boldsymbol{\mu}_h \in \text{int}(\mathcal{S})$ at random)

**repeat**

{The Assignment Step}

Set $\mathcal{X}_h \leftarrow \emptyset,\ h = 1, \cdots, k$

**for** $i = 1$ to $n$ **do**

$\mathcal{X}_h \leftarrow \mathcal{X}_h \cup \{\mathbf{x}_i\}$

where $h = h^\dagger(\mathbf{x}_i) = \operatorname{argmin}_{h'} d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_{h'})$

**end for**

{The Re-estimation Step}

**for** $h = 1$ to $k$ **do**

$\pi_h \leftarrow \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i$

$\boldsymbol{\mu}_h \leftarrow \sum_{\mathbf{x}_i \in \mathcal{X}_h} \frac{\nu_i}{\pi_h} \mathbf{x}_i$

**end for**

**until** *convergence*

*Proof:* The result follows since the algorithm monotonically decreases the objective function value, and the number of distinct clusterings is finite. ∎

In addition to local optimality, the Bregman hard clustering algorithm has the following interesting properties.

**Exhaustiveness:** The Bregman hard clustering algorithm with cluster centroids as optimal representatives works for *all* Bregman divergences and *only* for Bregman divergences since the centroid is the best predictor *only* for Bregman divergences [BGW05]. However, it is possible to have a similar alternate minimization based clustering algorithm for distance functions that are not Bregman divergences, the primary difference being that the optimal cluster representative, when it exists, will no longer be the mean or the expectation. The `convex-kmeans` clustering algorithm [MS03] and the generalizations of the

LBG algorithm [LBG80] are examples of such alternate minimization schemes where a (unique) representative exists because of convexity.

**Linear Separators:** For all Bregman divergences, the partitions induced by the Bregman hard clustering algorithm are separated by hyperplanes. In particular, the locus of points that are equidistant to two fixed points $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ in terms of a Bregman divergence is given by $\mathcal{X} = \{\mathbf{x} \mid d_\phi(\mathbf{x}, \boldsymbol{\mu}_1) = d_\phi(\mathbf{x}, \boldsymbol{\mu}_2)\}$, i.e., the set of points

$$\{\mathbf{x} \mid \langle \mathbf{x}, \nabla\phi(\boldsymbol{\mu}_2) - \nabla\phi(\boldsymbol{\mu}_1)\rangle = (\phi(\boldsymbol{\mu}_1) - \langle \boldsymbol{\mu}_1, \nabla\phi(\boldsymbol{\mu}_1)\rangle) - (\phi(\boldsymbol{\mu}_2) - \langle \boldsymbol{\mu}_2, \nabla\phi(\boldsymbol{\mu}_2)\rangle)\}\,,$$

which corresponds to a hyperplane.

**Scalability:** The computational complexity of each iteration of the Bregman hard clustering algorithm is linear in the number of data points and the number of desired clusters for *all* Bregman divergences, which makes the algorithm scalable and appropriate for large clustering problems.

**Applicability to mixed data types:** The Bregman hard clustering algorithm is applicable to mixed data types that are commonly encountered in machine learning. One can choose different convex functions that are appropriate and meaningful for different subsets of the features. The Bregman divergence corresponding to a convex combination of the component convex functions can then be used to cluster the data.

# Chapter 4

# Soft Clustering with Bregman Divergences

We now turn our attention to *soft* clustering with Bregman divergences. To this end, we first establish the fact that there is a uniquely determined Bregman divergence corresponding to every regular exponential family distribution. Later, we make this relation more precise by establishing a bijection between regular exponential families and *regular Bregman divergences*. The correspondence will be used to develop soft clustering algorithms with Bregman divergences.

## 4.1 Preliminaries

To present the correspondence and related results, we first review some background material on exponential families and Legendre duality in Sections 4.1.1 and 4.1.2.

[0]The work presented in this chapter has earlier appeared in part as [BMDG04].

### 4.1.1 Exponential families

Consider a measurable space $(\Omega, \mathcal{B})$ where $\mathcal{B}$ is a $\sigma$-algebra on the set $\Omega$. Let $\mathbf{t}$ be a measurable mapping from $\Omega$ to a set $\mathcal{T} \subseteq \mathbb{R}^d$, where $\mathcal{T}$ may be discrete (e.g., $\mathcal{T} \subset \mathbb{N}$). Let $p_0 : \mathcal{T} \mapsto \mathbb{R}_+$ be any function such that if $(\Omega, \mathcal{B})$ is endowed with a measure with density $dP_0(\omega) = p_0(\mathbf{t}(\omega))d\mathbf{t}(\omega)$, then $\int_{\omega\in\Omega} dP_0(\omega) < \infty$. The measure $P_0$ is absolutely continuous with respect to the Lebesgue measure $d\mathbf{t}(\omega)$. When $\mathcal{T}$ is a discrete set, $d\mathbf{t}(\omega)$ is the counting measure and $P_0$ is absolutely continuous with respect to the counting measure[1].

Now, $\mathbf{t}(\omega)$ is a random variable from $(\Omega, \mathcal{B}, P_0)$ to $(\mathcal{T}, \sigma(\mathcal{T}))$, where $\sigma(\mathcal{T})$ denotes the $\sigma$-algebra generated by $\mathcal{T}$. Let $\Theta$ be defined as the set of all parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ for which

$$\int_{\omega\in\Omega} \exp\left(\langle \boldsymbol{\theta}, \mathbf{t}(\omega)\rangle\right) dP_0(\omega) < \infty \ .$$

Based on the definition of $\Theta$, it is possible to define a function $\psi : \Theta \mapsto \mathbb{R}$ such that

$$\psi(\boldsymbol{\theta}) = \log\left(\int_{\omega\in\Omega} \exp(\langle \boldsymbol{\theta}, \mathbf{t}(\omega)\rangle) dP_0(\omega)\right) \ . \tag{4.1}$$

A family of probability distributions $\mathcal{F}_\psi$ parameterized by a $d$-dimensional vector $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d$ such that the probability density functions with respect to the measure $d\mathbf{t}(\omega)$ can be expressed in the form

$$f(\omega; \boldsymbol{\theta}) = \exp\left(\langle \boldsymbol{\theta}, \mathbf{t}(\omega)\rangle - \psi(\boldsymbol{\theta})\right) p_0(\mathbf{t}(\omega)) \tag{4.2}$$

is called an *exponential family* with *natural statistic* $\mathbf{t}(\omega)$, *natural parameter space* $\Theta$ and *natural parameter* $\boldsymbol{\theta}$. In particular, if the $\mathbf{t}(\omega)$ are affinely independent,

[1]For conciseness, we abuse notation and continue to use the Lebesgue integration sign even for counting measures. The integral in this case actually denotes a sum over $\mathcal{T}$. Further, the use of absolute continuity in the context of counting measure is non-standard. We say the measure $P_0$ is absolutely continuous with respect to the counting measure $\mu_c$ if $P_0(E) = 0$ for every set with $\mu_c(E) = 0$, where $E$ is a discrete set.

i.e., $\nexists$ non-zero $\mathbf{a} \in \mathbb{R}^d$ such that $\langle \mathbf{a}, \mathbf{t}(\omega) \rangle = c$, a constant $\forall \omega \in \Omega$[2], then this representation is said to be *minimal*. For a minimal representation, there exists a unique probability density $f(\omega; \boldsymbol{\theta})$ for every choice of $\boldsymbol{\theta} \in \Theta$ [WJ03]. $\mathcal{F}_\psi$ is called a *full exponential family* of *order* $d$ in such a case. In addition, if the parameter space $\Theta$ is open, i.e. $\Theta = \text{int}(\Theta)$, then $\mathcal{F}_\psi$ is called a *regular exponential family*.

It can be easily seen that if $\mathbf{x} \in \mathbb{R}^d$ denotes the natural statistic $\mathbf{t}(\omega)$, then the probability density function $g(\mathbf{x}; \boldsymbol{\theta})$ (with respect to the appropriate measure $d\mathbf{x}$) given by

$$g(\mathbf{x}; \boldsymbol{\theta}) = \exp(\langle \boldsymbol{\theta}, \mathbf{x} \rangle - \psi(\boldsymbol{\theta})) p_0(\mathbf{x}) \tag{4.3}$$

is such that $f(\omega; \boldsymbol{\theta}) / g(\mathbf{x}; \boldsymbol{\theta})$ does not depend on $\boldsymbol{\theta}$. Thus, $\mathbf{x}$ is a sufficient statistic [AN01] for the family, and in fact, can be shown [BN78] to be minimally sufficient. For our analysis, it is convenient to work with the minimal natural sufficient statistic $\mathbf{x}$ and hence, we redefine regular exponential families in terms of the probability density of $\mathbf{x} \in \mathbb{R}^d$, noting that the original probability space can actually be quite general.

**Definition 3** A multivariate parametric family $\mathcal{F}_\psi$ of distributions $\{p_{(\psi, \boldsymbol{\theta})} | \boldsymbol{\theta} \in \Theta = \text{int}(\Theta) = \text{dom}(\psi) \subseteq \mathbb{R}^d\}$ is called a regular exponential family if each probability density is of the form

$$p_{(\psi, \boldsymbol{\theta})}(\mathbf{x}) = \exp(\langle \mathbf{x}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta})) p_0(\mathbf{x}) \ .$$

The function $\psi(\boldsymbol{\theta})$ is known as the *log partition function* or the *cumulant function* corresponding to the exponential family. Given a regular exponential family $\mathcal{F}_\psi$, the log-partition function $\psi$ is uniquely determined up to a constant additive term. It can be shown [BN78] that $\Theta$ is a non-empty convex set in $\mathbb{R}^d$ and $\psi$ is a convex function. In fact, it is possible to prove a stronger result that characterizes $\psi$ in

[2]Strictly speaking, $\nexists \mathbf{a} \neq 0$ such that $P_0(\{\omega : \langle \mathbf{t}(\omega), \mathbf{a} \rangle = c\}) = 1$.

terms of a special class of convex functions called Legendre functions, which are defined below.

**Definition 4 ([Roc70])** Let $\psi$ be a proper, closed[3] convex function with $\Theta = \text{int}(\text{dom}(\psi))$. The pair $(\Theta, \psi)$ is called a convex function of Legendre type or a Legendre function if the following properties are satisfied:

(L1) $\Theta$ is non-empty,

(L2) $\psi$ is strictly convex and differentiable on $\Theta$,

(L3) $\forall \boldsymbol{\theta}_b \in \text{bd}(\Theta), \lim_{\boldsymbol{\theta} \to \boldsymbol{\theta}_b} \|\nabla \psi(\boldsymbol{\theta})\| \to \infty$ where $\boldsymbol{\theta} \in \Theta$.

Based on this definition, we now state a critical property of the cumulant function of regular exponential families.

**Lemma 1** *Let $\psi$ be the cumulant function of a regular exponential family with natural parameter space $\Theta = \text{dom}(\psi)$. Then, $\psi$ is a proper closed convex function with $\text{int}(\Theta) = \Theta$ and $(\Theta, \psi)$ is a convex function of Legendre type.*

The above result directly follows from Theorems 8.2, 9.1 and 9.3 of [BN78].

### 4.1.2 Expectation parameters and Legendre duality

Consider a $d$-dimensional real random vector $X$ distributed according to a regular exponential family density $p_{(\psi,\boldsymbol{\theta})}$ specified by the natural parameter $\boldsymbol{\theta} \in \Theta$. The expectation of $X$ with respect to $p_{(\psi,\boldsymbol{\theta})}$, also called the *expectation parameter*, is given by

$$\boldsymbol{\mu} = \boldsymbol{\mu}(\boldsymbol{\theta}) = E_{p_{(\psi,\boldsymbol{\theta})}}[X] = \int_{\mathbb{R}^d} \mathbf{x} p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}) d\mathbf{x}. \tag{4.4}$$

[3]A convex function $\psi$ is proper if $\text{dom}(\psi)$ is non-empty and $\forall \mathbf{x} \in \text{dom}(\psi), \psi(\mathbf{x}) > -\infty$. A convex function is closed if it is lower semi-continuous.

It can be shown [BN78, Ama95] that the expectation and natural parameters have a one-one correspondence with each other and span spaces that exhibit a dual relationship. To specify the duality more precisely, we first define conjugate functions.

**Definition 5 ([Roc70])** Let $\psi$ be a real-valued function on $\mathbb{R}^d$. Then its **conjugate function** $\psi^*$ is given by

$$\psi^*(\mathbf{t}) = \sup_{\boldsymbol{\theta} \in \mathbb{R}^d} \{\langle \mathbf{t}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta})\}. \tag{4.5}$$

Conjugates of convex functions satisfy the following useful result.

**Theorem 2 ([BN78])** *If $\psi$ is a proper closed convex function, $\psi^*$ is also a proper closed convex function and $\psi^{**} = \psi$.*

When $\psi$ is strictly convex and differentiable over $\Theta = \text{int}(\text{dom}(\psi))$, we can obtain the unique $\boldsymbol{\theta}^\dagger$ that corresponds to the supremum in (4.5) by setting the gradient of $\langle \mathbf{t}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta})$ to zero, i.e.,

$$\nabla(\langle \mathbf{t}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta}))\,\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^\dagger} = 0 \qquad \text{and so} \qquad \mathbf{t} = \nabla\psi(\boldsymbol{\theta}^\dagger)\ . \tag{4.6}$$

The strict convexity of $\psi$ implies that $\nabla\psi$ is monotonic and it is possible to define the inverse function $(\nabla\psi)^{-1} : \Theta^* \mapsto \Theta$, where $\Theta^* = \text{int}(\text{dom}(\psi^*))$. If the pair $(\Theta, \psi)$ is of Legendre type, then it can be shown [Roc70, BN78] that $(\Theta^*, \psi^*)$ is also of Legendre type and $(\Theta, \psi)$ and $(\Theta^*, \psi^*)$ are called Legendre duals of each other. Further, the gradient mappings are continuous and form a bijection between the two open sets $\Theta$ and $\Theta^*$. The relation between $(\Theta, \psi)$ and $(\Theta^*, \psi^*)$ is formally stated below.

**Theorem 3 ([Roc70])** *Let $\psi$ be a real-valued proper closed convex function with conjugate function $\psi^*$. Let $\Theta = \text{int}(\text{dom}(\psi))$ and $\Theta^* = \text{int}(\text{dom}(\psi^*))$. If $(\Theta, \psi)$ is a convex function of Legendre type, then*

*(i)* $(\Theta^*, \psi^*)$ *is a convex function of Legendre type,*

*(ii)* $(\Theta, \psi)$ *and* $(\Theta^*, \psi^*)$ *are Legendre duals of each other,*

*(iii) The gradient function* $\nabla\psi : \Theta \mapsto \Theta^*$ *is a one-to-one function from the open convex set* $\Theta$ *onto the open convex set* $\Theta^*$*,*

*(iv) The gradient functions* $\nabla\psi, \nabla\psi^*$ *are continuous, and* $\nabla\psi^* = (\nabla\psi)^{-1}$*.*

Let us now look at the relationship between the natural parameter $\boldsymbol{\theta}$ and the expectation parameter $\boldsymbol{\mu}$ defined in (4.4). Differentiating the identity $\int p_{(\psi,\boldsymbol{\theta})}(\mathbf{x})d\mathbf{x} = 1$ with respect to $\boldsymbol{\theta}$ gives us $\boldsymbol{\mu} = \boldsymbol{\mu}(\boldsymbol{\theta}) = \nabla\psi(\boldsymbol{\theta})$, i.e., the expectation parameter $\boldsymbol{\mu}$ is the image of the natural parameter $\boldsymbol{\theta}$ under the gradient mapping $\nabla\psi$. Let $\phi$ denote the conjugate of $\psi$, i.e.,

$$\phi(\boldsymbol{\mu}) = \sup_{\boldsymbol{\theta}}\{\langle\boldsymbol{\mu}, \boldsymbol{\theta}\rangle - \psi(\boldsymbol{\theta})\}. \tag{4.7}$$

Since $(\Theta, \psi)$ is a convex function of Legendre type (Lemma 1), the pairs $(\Theta, \psi)$ and $(\text{int}(\text{dom}(\phi)), \phi)$ are Legendre duals of each other from Theorem 3, i.e., $\phi = \psi^*$ and $\text{int}(\text{dom}(\phi)) = \Theta^*$. Thus, the mappings between the dual spaces $\text{int}(\text{dom}(\phi))$ and $\Theta$ are given by the Legendre transformation

$$\boldsymbol{\mu}(\boldsymbol{\theta}) = \nabla\psi(\boldsymbol{\theta}) \qquad \text{and} \qquad \boldsymbol{\theta}(\boldsymbol{\mu}) = \nabla\phi(\boldsymbol{\mu}) \ . \tag{4.8}$$

Further, the conjugate function $\phi$ can be expressed as

$$\phi(\boldsymbol{\mu}) = \langle\boldsymbol{\theta}(\boldsymbol{\mu}), \boldsymbol{\mu}\rangle - \psi(\boldsymbol{\theta}(\boldsymbol{\mu})), \quad \forall\boldsymbol{\mu} \in \text{int}(\text{dom}(\phi)) \ . \tag{4.9}$$

### 4.1.3 Exponential families and Bregman divergences

We are now ready to explicitly state the formal connection between exponential families of distributions and Bregman divergences. It has been observed in the literature

that exponential families and Bregman divergences have a close relationship that can be exploited for several learning problems. In particular, [FW00, Section 5.1] remarked that the log-likelihood of the density of an exponential family distribution $p_{(\psi,\boldsymbol{\theta})}$ can be written as the sum of the negative of a uniquely determined Bregman divergence $d_\phi(\mathbf{x}, \boldsymbol{\mu})$ and a function that does not depend on the distribution parameters. In our notation, this can be written as

$$\log(p_{(\psi,\boldsymbol{\theta})}(\mathbf{x})) = -d_\phi(\mathbf{x}, \boldsymbol{\mu}(\boldsymbol{\theta})) + \log(b_\phi(\mathbf{x})) \ , \tag{4.10}$$

where $\phi$ is the convex conjugate of $\psi$ and $\boldsymbol{\mu} = \boldsymbol{\mu}(\boldsymbol{\theta}) = \nabla\psi(\boldsymbol{\theta})$ is the expectation parameter corresponding to $\boldsymbol{\theta}$. The result was later used by [CDS01] to extend PCA to exponential families. However, as we explain below, a formal proof is required to show that (4.10) holds for all $\mathbf{x}$ of interest. We focus on the case when $p_{(\psi,\boldsymbol{\theta})}$ is a *regular* exponential family.

To get an intuition of the result, observe that the log-likelihood of any exponential family, considering only the parametric terms, can be written as

$$\begin{aligned} \langle \mathbf{x}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta}) &= (\langle \boldsymbol{\mu}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta})) + \langle \mathbf{x} - \boldsymbol{\mu}, \boldsymbol{\theta} \rangle \\ &= \phi(\boldsymbol{\mu}) + \langle \mathbf{x} - \boldsymbol{\mu}, \nabla\phi(\boldsymbol{\mu}) \rangle \ , \end{aligned}$$

from (4.8) and (4.9), where $\boldsymbol{\mu} \in \mathrm{int}(\mathrm{dom}(\phi))$. Therefore, for any $\mathbf{x} \in \mathrm{dom}(\phi)$, $\boldsymbol{\theta} \in \Theta$, and $\boldsymbol{\mu} \in \mathrm{int}(\mathrm{dom}(\phi))$, one can write

$$\langle \mathbf{x}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta}) \ = \ - d_\phi(\mathbf{x}, \boldsymbol{\mu}) + \phi(\mathbf{x}) \ .$$

Then considering the density of an exponential family with respect to the appropriate measure $d\mathbf{x}$, we have

$$\log(p_{(\psi,\boldsymbol{\theta})}(\mathbf{x})) \ = \ \langle \mathbf{x}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta}) + \log p_0(\mathbf{x}) \ = \ - d_\phi(\mathbf{x}, \boldsymbol{\mu}) + \log(b_\phi(\mathbf{x})) \ ,$$

where $b_\phi(\mathbf{x}) = \exp(\phi(\mathbf{x}))p_0(\mathbf{x})$.

Thus (4.10) follows directly from Legendre duality for $\mathbf{x} \in \text{dom}(\phi)$. However, for (4.10) to be useful, one would like to ensure that it is true for all individual "instances" $\mathbf{x}$ that can be drawn following the exponential distribution $p_{(\psi,\boldsymbol{\theta})}$. Let $I_\psi$ denote the set of such instances. Establishing (4.10) can be tricky for $\mathbf{x} \in I_\psi$ since the relationship between $I_\psi$ and $\text{dom}(\phi)$ is not apparent. Further, there are distributions for which the instances space $I_\psi$ and the expectation parameter space $\text{int}(\text{dom}(\phi))$ are disjoint, as the following example shows.

**Example 8** A Bernoulli random variable $X$ takes values in $\{0, 1\}$ such that $p(X = 1) = \pi$ and $p(X = 0) = 1 - \pi$, for some $\pi \in [0, 1]$. The instance space for $X$ is just $I_\psi = \{0, 1\}$. The cumulant function for $X$ is $\psi(\theta) = \log(1 + \exp(\theta))$ with $\Theta = \mathbb{R}$. A simple calculation shows that the conjugate function $\phi(\mu) = \mu \log \mu + (1-\mu) \log(1-\mu),\ \forall \mu \in (0, 1)$. Since $\phi$ is a closed function, we obtain $\phi(\mu) = 0$ for $\mu \in \{0, 1\}$ by taking the limits. Thus, the effective domain of $\phi$ is $\text{dom}(\phi) = [0, 1]$ and $\pi = \mu$, whereas the expectation parameter space is given by $\text{int}(\text{dom}(\phi)) = (0, 1)$. Hence the instance space $I_\psi$ and the expectation parameter space $\text{int}(\text{dom}(\phi))$ are disjoint; however $I_\psi \subset \text{dom}(\phi)$.

In this particular case, since the "instances" lie within $\text{dom}(\phi)$, the relation (4.10) does hold for all $\mathbf{x} \in I_\psi$. However, it remains to be shown that $I_\psi \subseteq \text{dom}(\phi)$ for all regular exponential family distributions.

In order to establish such a result for all regular exponential family distributions, we need to formally define the set of instances $I_\psi$. If the measure $P_0$ is absolutely continuous with respect to the counting measure, then $\mathbf{x} \in I_\psi$ if $p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}) > 0$. On the other hand, if $P_0$ is absolutely continuous with respect to the Lebesgue measure, then $\mathbf{x} \in I_\psi$ if all sets with positive Lebesgue measure that contain $\mathbf{x}$ have positive probability mass. Now, a closer look reveals that the set of instances $I_\psi$ is independent of the choice of $\boldsymbol{\theta}$. In fact, $I_\psi$ is just the support of $P_0$ and can be

formally defined as follows.

**Definition 6** Let $I_\psi$ denote the set of instances that can be drawn following $p_{(\psi,\boldsymbol{\theta})}(\mathbf{x})$. Then, $\mathbf{x} \in I_\psi$ if $\forall S$ such that $\mathbf{x} \in S$ and $\int_S d\mathbf{x} > 0$, we have $\int_S dP_0(\mathbf{x}) > 0$, where $P_0$ is as defined in Section 4.1.1.

The following theorem establishes that the set of instances $I_\psi$ is always a subset of $\mathrm{dom}(\phi)$.

**Theorem 4** *Let $I_\psi$ be the set of instances as in Definition 6. Then, $I_\psi \subseteq \mathrm{dom}(\phi)$ where $\phi$ is the convex conjugate of $\psi$.*

The theorem follows from Theorem 9.1 and related results in [BN78]. Since the proof of Theorem 4 is technical, we have relegated it and related results to Appendix B.

We are now ready to formally state and prove the relationship between exponential family distributions and Bregman divergences. Note that it is sufficient to establish the relationship for all $\mathbf{x} \in \mathrm{dom}(\phi)$ since Theorem 4 establishes that $I_\psi \subseteq \mathrm{dom}(\phi)$, thereby ensuring that the relationship holds for all individual instances $\mathbf{x} \in I_\psi$ that can be drawn following the regular exponential family distribution $p_{(\psi,\boldsymbol{\theta})}$.

**Theorem 5** *Let $p_{(\psi,\boldsymbol{\theta})}$ be the probability density function of a regular exponential family distribution. Let $\phi$ be the convex conjugate of $\psi$ so that $(\mathrm{int}(\mathrm{dom}(\phi)), \phi)$ is the Legendre dual of $(\Theta, \psi)$. Let $\boldsymbol{\theta} \in \Theta$ be the natural parameter and $\boldsymbol{\mu} \in \mathrm{int}(\mathrm{dom}(\phi))$ be the corresponding expectation parameter. Let $d_\phi$ be the Bregman divergence derived from $\phi$. Then,*

$$p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}) = \exp(-d_\phi(\mathbf{x}, \boldsymbol{\mu}))b_\phi(\mathbf{x}), \qquad \forall \mathbf{x} \in \mathrm{dom}(\phi) \tag{4.11}$$

*where $b_\phi : \mathrm{dom}(\phi) \mapsto \mathbb{R}_+$ is a uniquely determined function.*

*Proof:* For all $\mathbf{x} \in \text{dom}(\phi)$, we have

$$\begin{aligned} p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}) &= \exp(\langle \mathbf{x}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta}))p_0(\mathbf{x}) \\ &= \exp(\phi(\boldsymbol{\mu}) + \langle \mathbf{x} - \boldsymbol{\mu}, \nabla\phi(\boldsymbol{\mu}) \rangle)p_0(\mathbf{x}) \qquad \text{(using (4.8) and (4.9))} \\ &= \exp(-\{\phi(\mathbf{x}) - \phi(\boldsymbol{\mu}) - \langle \mathbf{x} - \boldsymbol{\mu}, \nabla\phi(\boldsymbol{\mu}) \rangle\} + \phi(\mathbf{x}))p_0(\mathbf{x}) \\ &= \exp(-d_\phi(\mathbf{x}, \boldsymbol{\mu}))b_\phi(\mathbf{x}) \ , \end{aligned}$$

where $b_\phi(\mathbf{x}) = \exp(\phi(\mathbf{x}))p_0(\mathbf{x}))$.

We observe that $p_{(\psi,\boldsymbol{\theta})}$ uniquely determines the log-partition function $\psi$ to a constant additive term so that the gradient space of all the possible functions $\psi$ is the same, i.e., the expectation parameter $\boldsymbol{\mu} = \nabla\psi(\boldsymbol{\theta})$ corresponding to $\boldsymbol{\theta}$ is uniquely determined and the corresponding conjugate functions $\phi$ differ only by a constant additive term. Hence the Bregman divergence $d_\phi(\mathbf{x}, \boldsymbol{\mu})$ derived from any of these conjugate functions will be identical since constant additive terms do not change the corresponding Bregman divergence (Appendix A, Property 4). The Legendre duality between $\phi$ and $\psi$ also ensures that no two exponential families correspond to the same Bregman divergence, i.e., the mapping is one-to-one. Further, since $p_{(\psi.\boldsymbol{\theta})}(\mathbf{x})$ is well-defined on $\text{dom}(\phi)$, and the corresponding $d_\phi(\mathbf{x}, \boldsymbol{\mu})$ is unique, the function $b_\phi(\mathbf{x}) = \exp(d_\phi(\mathbf{x}, \boldsymbol{\mu}))p_{(\psi,\boldsymbol{\theta})}(\mathbf{x})$ is uniquely determined. ■

### 4.1.4 Bijection with regular Bregman divergences

From Theorem 5 we note that every regular exponential family corresponds to a unique and distinct Bregman divergence (one-to-one mapping). Now, we investigate whether there is a regular exponential family corresponding to every choice of Bregman divergence (onto mapping).

For regular exponential families, the cumulant function $\psi$ as well as its conjugate $\phi$ are convex functions of Legendre type. Hence, for a Bregman divergence

generated from a convex function $\phi$ to correspond to a regular exponential family, it is necessary that $\phi$ be of Legendre type. Further, it is necessary that the Legendre conjugate $\psi$ of $\phi$ to be $C^\infty$, since cumulant functions of regular exponential families are $C^\infty$. However, it is not clear if these conditions are sufficient. Instead, we provide a sufficiency condition using exponentially convex functions [Akh65, EGG03], which are defined below.

**Definition 7** A function $f : \Theta \mapsto \mathbb{R}_{++},\ \Theta \subseteq \mathbb{R}^d$ is called exponentially convex if the kernel $K_f(\alpha, \beta) = f(\alpha + \beta)$, with $\alpha + \beta \in \Theta$, satisfies

$$\sum_{i=1}^{n}\sum_{j=1}^{n} K_f(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) u_i \bar{u}_j \geq 0$$

for any set $\{\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_n\} \subseteq \Theta$ with $\boldsymbol{\theta}_i + \boldsymbol{\theta}_j \in \Theta \forall i, j$, and $\{u_1, \cdots, u_n\} \subset \mathbb{C}$ ($\bar{u}_j$ denotes the complex conjugate of $u_j$), i.e, the kernel $K_f$ is positive semi-definite.

Although it is well known that the logarithm of an exponentially convex function is a convex function [Akh65], we are interested in the case where the logarithm is strictly convex with an open domain. Using this class of exponentially convex functions, we now define a class of Bregman divergences called *regular Bregman divergences*.

**Definition 8** Let $f : \Theta \mapsto \mathbb{R}_{++}$ be a continuous exponentially convex function such that $\Theta$ is open and $\psi(\boldsymbol{\theta}) = \log(f(\boldsymbol{\theta}))$ is strictly convex. Let $\phi$ be the conjugate function of $\psi$. Then we say that the Bregman divergence $d_\phi$ derived from $\phi$ is a *regular Bregman divergence*.

We will now prove that there is a bijection between regular exponential families and regular Bregman divergences. The crux of the argument relies on results in harmonic analysis connecting positive definiteness to integral transforms [BCR84]. In particular, we use a result due to [Dev55] that relates exponentially convex functions to Laplace transforms of bounded non-negative measures.

**Theorem 6 ([Dev55])** *Let $\Theta \subseteq \mathbb{R}^d$ be an open convex set. A necessary and sufficient condition that there exists a unique, bounded, non-negative measure $\nu$ such that $f : \Theta \mapsto \mathbb{R}_+$ can be represented as*

$$f(\boldsymbol{\theta}) = \int_{\mathbf{x} \in \mathbb{R}^d} \exp(\langle \mathbf{x}, \boldsymbol{\theta} \rangle) d\nu(\mathbf{x}) \tag{4.12}$$

*is that $f$ is continuous and exponentially convex.*

There are two parts to the argument leading to the bijection result. Note that we have already established in Theorem 5 that there is a unique Bregman divergence corresponding to every exponential family distribution. In Lemma 2 we show that these Bregman divergences are regular (one-to-one). Further, in Lemma 3, we show that there exists a unique regular exponential family determined by every regular Bregman divergence (onto).

**Lemma 2** *Let $\mathcal{F}_\psi$ be a regular exponential family with cumulant function $\psi$ and natural parameter space $\Theta$. Let $\phi$ be the conjugate of $\psi$. Then $d_\phi$ is a regular Bregman divergence.*

*Proof:* Since $\mathcal{F}_\psi$ is a regular exponential family, there exists a non-negative bounded measure $\nu$ such that for all $\boldsymbol{\theta} \in \Theta$,

$$\begin{aligned} 1 &= \int_{\mathbf{x} \in \mathbb{R}^d} \exp(\langle \mathbf{x}, \boldsymbol{\theta} \rangle - \psi(\boldsymbol{\theta})) d\nu(\mathbf{x}) \\ \text{and so} \qquad \exp(\psi(\boldsymbol{\theta})) &= \int_{\mathbf{x} \in \mathbb{R}^d} \exp(\langle \mathbf{x}, \boldsymbol{\theta} \rangle) d\nu(\mathbf{x}). \end{aligned}$$

Thus, from Theorem 6, $\exp(\psi(\boldsymbol{\theta}))$ is a continuous exponentially convex function with the open set $\Theta$ as its domain. Further, being the cumulant of a regular exponential family, $\psi$ is strictly convex. Therefore, the Bregman divergence $d_\phi$ derived from the conjugate function $\phi$ of $\psi$ is a regular Bregman divergence. ∎

**Lemma 3** *Let $d_\phi$ be a regular Bregman divergence generated by the convex function $\phi$. Let $(\Theta, \psi)$ be the Legendre dual of $(\mathrm{int}(\mathrm{dom}(\phi)), \phi)$ where $\Theta = \mathrm{int}(\mathrm{dom}(\psi))$. Then there exists a unique regular exponential family $\mathcal{F}_\psi$ whose natural parameter space is $\Theta$ and cumulant function equals $\psi$ (upto a constant additive term).*

We first need the following result for the proof of Lemma 3.

**Lemma 4** *Let $\psi$ be the cumulant of an exponential family with base measure $P_0$ and natural parameter space $\Theta \subseteq \mathbb{R}^d$. Then $\psi$ is not strictly convex if and only if $P_0$ is concentrated on an affine subspace of $\mathbb{R}^d$.*

*Proof:* First we focus on the 'only if' part. If $\psi$ is not strictly convex, then there must exist a set $I \subseteq \mathbb{R}^d$ such that for any $\boldsymbol{\theta} \in I$, $\psi(\boldsymbol{\theta}) = c + \langle \mathbf{a}, \boldsymbol{\theta} \rangle$. Hence, $\exp(\psi(\boldsymbol{\theta})) = \exp(c + \langle \mathbf{a}, \boldsymbol{\theta} \rangle)$. Since $\psi(\boldsymbol{\theta})$ is the cumulant of an exponential family with base measure $P_0(\mathbf{x})$, $\forall \theta \in I$ we have

$$\begin{aligned} & \int_{\mathbf{x} \in \mathbb{R}^d} \exp(\langle \mathbf{x}, \boldsymbol{\theta} \rangle) dP_0(\mathbf{x}) \;=\; \exp(\psi(\boldsymbol{\theta})) \;=\; \exp(c + \langle \mathbf{a}, \boldsymbol{\theta} \rangle) \\ \Rightarrow \quad & \int_{\mathbf{x} \in \mathbb{R}^d} \exp(\langle \mathbf{x} - \mathbf{a}, \boldsymbol{\theta} \rangle) dP_0(\mathbf{x}) \;=\; \exp(c) \;, \end{aligned}$$

which is possible only if $P_0(\mathbf{x})$ is concentrated on the affine subspace $\{\mathbf{x} \in \mathbb{R}^d | \langle \mathbf{x} - \mathbf{a}, \boldsymbol{\theta} \rangle = \text{constant}\}$ for $\boldsymbol{\theta} \in I$.

Now, we prove the 'if' part. Say $P_0(\mathbf{x})$ is concentrated on an affine subspace $S = \{\mathbf{x} \in \mathbb{R}^d | \langle \mathbf{x}, \mathbf{b} \rangle = c\}$ for some $\mathbf{b} \in \mathbb{R}^d$ and constant $c$. Let $I = \{\boldsymbol{\theta} | \boldsymbol{\theta} = \alpha \mathbf{b},\ \alpha \in \mathbb{R}\}$. Then, for any $\boldsymbol{\theta} = \alpha \mathbf{b} \in I$, we have $\langle \mathbf{x}, \boldsymbol{\theta} \rangle = \alpha c\ \forall \mathbf{x} \in S$ and the cumulant is given by

$$\begin{aligned} \psi(\boldsymbol{\theta}) \;&=\; \log\left( \int_{\mathbf{x} \in \mathbb{R}^d} \exp(\langle \mathbf{x}, \boldsymbol{\theta} \rangle) dP_0(\mathbf{x}) \right) \;=\; \log\left( \int_{\mathbf{x} \in S} \exp(\langle \mathbf{x}, \boldsymbol{\theta} \rangle) dP_0(\mathbf{x}) \right) \\ &=\; \log\left( \int_{\mathbf{x} \in S} \exp(\alpha c) dP_0(\mathbf{x}) \right) \;=\; \log(\exp(\alpha c) P_0(S)) \;=\; \alpha c + \log(P_0(S)) \\ &=\; \langle \mathbf{x}_0, \boldsymbol{\theta} \rangle + \log(P_0(S)) \;, \end{aligned}$$

for any $\mathbf{x}_0 \in S$, implying that $\psi$ is not strictly convex. ∎

**Proof of Lemma 3** Let the regular Bregman divergence $d_\phi$ be generated by $\phi$ and let $\psi$ be the conjugate of $\phi$. Since $d_\phi$ is a regular Bregman divergence, by definition 8, $\psi$ is strictly convex with $\text{dom}(\psi) = \Theta$ being an open set. Further, the function $\exp(\psi(\boldsymbol{\theta}))$ is a continuous, exponentially convex function. From Theorem 6, there exists a unique non-negative bounded measure $\nu$ that satisfies (4.12). Without any loss of generality, we assume $\mathbf{0} \in \Theta$ so that[4]

$$\exp(\psi(\mathbf{0})) = \int_{\mathbf{x}\in\mathbb{R}^d} d\nu(\mathbf{x})$$

and $P_0(\mathbf{x}) = \nu(\mathbf{x})/\exp(\psi(\mathbf{0}))$ is a probability measure. The set of all $\boldsymbol{\theta} \in \mathbb{R}^d$ for which

$$\int_{\mathbf{x}\in\mathbb{R}^d} \exp(\langle\boldsymbol{\theta}, \mathbf{x}\rangle) dP_0(\mathbf{x}) < \infty$$

is same as the set $\{\boldsymbol{\theta} \in \mathbb{R}^d | \exp(\psi(\boldsymbol{\theta}) - \psi(\mathbf{0})) < \infty\}$, which is just $\Theta$ itself. Hence, the exponential family $\mathcal{F}_\psi$ consisting of densities of the form

$$p_{(\psi,\boldsymbol{\theta})} = \exp(\langle\boldsymbol{\theta}, \mathbf{x}\rangle - \psi(\boldsymbol{\theta}) + \psi(\mathbf{0}))$$

with respect to the measure $P_0(\mathbf{x})$ has $\Theta$ as its natural parameter space and $\psi(\boldsymbol{\theta}) - \psi(\mathbf{0})$ as the cumulant function.

Since $\psi$ is strictly convex on $\Theta$, it follows from Lemma 4 that the measure $P_0$ is not concentrated in an affine subspace of $\mathbb{R}^d$, i.e., $\mathbf{x}$ is a minimal statistic for $\mathcal{F}_\psi$. Therefore, the exponential family generated by $P_0$ and $\mathbf{x}$ is full. Since $\Theta$ is also open, it follows that $\mathcal{F}_\psi$ is a regular exponential family and the construction ensures

[4]We could have normalized $\nu$ to $P_0$ using $\exp(\psi(\mathbf{b}))$ for any $\mathbf{b} \in \Theta$ since $\Theta$ is non-empty.

Table 4.1: Various functions of interest for some popular exponential distributions. For all the cases shown in the table, $\mathbf{x}$ is the sufficient statistic. Note that for the Gaussian examples the variance $\sigma$ is assumed to be constant. The number of trials, $N$, for the binomial and multinomial examples is also assumed to be constant.

| **Distribution** | $p(\mathbf{x};\boldsymbol{\theta})$ | $\boldsymbol{\theta}$ | $\psi(\boldsymbol{\theta})$ | $\Theta$ |
|---|---|---|---|---|
| 1-D Gaussian | $\frac{1}{\sqrt{(2\pi\sigma^2)}}\exp(-\frac{(x-a)^2}{2\sigma^2})$ | $\frac{a}{\sigma^2}$ | $\frac{\sigma^2}{2}\theta^2$ | $\mathbb{R}$ |
| 1-D Poisson | $\frac{\lambda^x e^{-\lambda}}{x!}$ | $\log\lambda$ | $\exp(\theta)$ | $\mathbb{R}$ |
| 1-D Bernoulli | $q^x(1-q)^{1-x}$ | $\log(\frac{q}{1-q})$ | $\log(1+\exp(\theta))$ | $\mathbb{R}$ |
| 1-D Binomial | $\frac{N!}{(x)!(N-x)!}q^x(1-q)^{N-x}$ | $\log(\frac{q}{1-q})$ | $N\log(1+\exp(\theta))$ | $\mathbb{R}$ |
| 1-D Exponential | $\lambda\exp(-\lambda x)$ | $-\lambda$ | $-\log(-\theta)$ | $\mathbb{R}_{--}$ |
| $d$-D Sph. Gaussian | $\frac{1}{\sqrt{(2\pi\sigma^2)^d}}\exp(-\frac{\|\mathbf{x}-\mathbf{a}\|^2}{2\sigma^2})$ | $\frac{\mathbf{a}}{\sigma^2}$ | $\frac{\sigma^2}{2}\|\boldsymbol{\theta}\|^2$ | $\mathbb{R}^d$ |
| $d$-D Multinomial | $\frac{N!}{\prod_{j=1}^d (x_j)!}\prod_{j=1}^d (q_j)^{x_j}$ | $[\log(\frac{q_j}{q_d})]_{j=1}^{d-1}$ | $N\log(1+\sum_{j=1}^{d-1}\exp(\theta_j))$ | $\mathbb{R}^{d-1}$ |

| **Distribution** | $\boldsymbol{\mu}$ | $\phi(\boldsymbol{\mu})$ | $d_\phi(\mathbf{x},\boldsymbol{\mu})$ | $\mathrm{dom}(\phi)$ |
|---|---|---|---|---|
| 1-D Gaussian | $a$ | $\frac{1}{2\sigma^2}\mu^2$ | $\frac{1}{2\sigma^2}(x-\mu)^2$ | $\mathbb{R}$ |
| 1-D Poisson | $\lambda$ | $\mu\log\mu-\mu$ | $x\log(\frac{x}{\mu})-(x-\mu)$ | $\mathbb{R}_+$ |
| 1-D Bernoulli | $q$ | $\mu\log\mu+(1-\mu)\log(1-\mu)$ | $x\log(\frac{x}{\mu})+(1-x)\log(\frac{1-x}{1-\mu})$ | $[0,1]$ |
| 1-D Binomial | $Nq$ | $\mu\log(\frac{\mu}{N})+(N-\mu)\log(\frac{N-\mu}{N})$ | $x\log(\frac{x}{\mu})+(N-x)\log(\frac{N-x}{N-\mu})$ | $[0,N]$ |
| 1-D Exponential | $1/\lambda$ | $-\log\mu-1$ | $\frac{x}{\mu}-\log\left(\frac{x}{\mu}\right)-1$ | $\mathbb{R}_+$ |
| $d$-D Sph. Gaussian | $\mathbf{a}$ | $\frac{1}{2\sigma^2}\|\boldsymbol{\mu}\|^2$ | $\frac{1}{2\sigma^2}\|\mathbf{x}-\boldsymbol{\mu}\|^2$ | $\mathbb{R}^d$ |
| $d$-D Multinomial | $[Nq_j]_{j=1}^{d-1}$ | $\sum_{j=1}^d \mu_j\log(\frac{\mu_j}{N})$ | $\sum_{j=1}^d x_j\log(\frac{x_j}{\mu_j})$ | $\mathbb{R}^{d-1}$ |

that it is unique since $\psi$ can only differ by an additive constant. ∎

Lemmas 2 and 3 imply the desired bijection result.

**Theorem 7** *There is a bijection between regular exponential families and regular Bregman divergences.*

### 4.1.5 Examples

Table 4.1 shows the various functions of interest for some popular exponential distribution families. We now look at two common exponential families in detail and obtain the corresponding Bregman divergences.

**Example 9** The most well-known exponential family is that of Gaussian distribu-

tions, in particular uniform variance, spherical Gaussian distributions with densities of the form

$$p(\mathbf{x};\mathbf{a}) = \frac{1}{\sqrt{(2\pi\sigma^2)^d}} \exp(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{a}\|^2) \ ,$$

where $\mathbf{x}, \mathbf{a} \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}$ is a constant. As shown below, $p(\mathbf{x}, \mathbf{a})$ can be expressed in the canonical form for exponential families with natural parameter $\boldsymbol{\theta} = \frac{\mathbf{a}}{\sigma^2}$ and cumulant function $\psi(\boldsymbol{\theta}) = \frac{\sigma^2}{2}\|\boldsymbol{\theta}\|^2$,

$$\begin{aligned} p(\mathbf{x};\mathbf{a}) &= \frac{1}{\sqrt{(2\pi\sigma^2)^d}} \exp(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{a}\|^2) \\ &= \exp(\langle \mathbf{x}, \frac{\mathbf{a}}{\sigma^2}\rangle - \frac{1}{2\sigma^2}\|\mathbf{a}\|^2 - \frac{1}{2\sigma^2}\|\mathbf{x}\|^2)\frac{1}{\sqrt{(2\pi\sigma^2)^d}} \\ &= \exp(\langle \mathbf{x}, \boldsymbol{\theta}\rangle - \frac{\sigma^2}{2}\|\boldsymbol{\theta}\|^2)\exp(-\frac{1}{2\sigma^2}\|\mathbf{x}\|^2)\frac{1}{\sqrt{(2\pi\sigma^2)^d}} \\ &= \exp(\langle \mathbf{x}, \boldsymbol{\theta}\rangle - \psi(\boldsymbol{\theta}))p_0(\mathbf{x}) \ , \end{aligned}$$

where $p_0(\mathbf{x})$ is independent of $\boldsymbol{\theta}$. By (4.8), the expectation parameter for this distribution is given by

$$\boldsymbol{\mu} = \nabla\psi(\boldsymbol{\theta}) = \nabla(\frac{\sigma^2}{2}\|\boldsymbol{\theta}\|^2) = \sigma^2\boldsymbol{\theta} = \mathbf{a} \ .$$

By using (4.9), the Legendre conjugate function $\phi$ of $\psi$ is

$$\phi(\boldsymbol{\mu}) = \langle \boldsymbol{\mu}, \boldsymbol{\theta}\rangle - \psi(\boldsymbol{\theta}) = \langle \boldsymbol{\mu}, \frac{\boldsymbol{\mu}}{\sigma^2}\rangle - \frac{\sigma^2}{2}\|\boldsymbol{\theta}\|^2 = \frac{\|\boldsymbol{\mu}\|^2}{2\sigma^2} \ .$$

The corresponding Bregman divergence equals

$$\begin{aligned} d_\phi(\mathbf{x},\boldsymbol{\mu}) &= \phi(\mathbf{x}) - \phi(\boldsymbol{\mu}) - \langle \mathbf{x}-\boldsymbol{\mu}, \nabla\phi(\boldsymbol{\mu})\rangle \ = \ \frac{\|\mathbf{x}\|^2}{2\sigma^2} - \frac{\|\boldsymbol{\mu}\|^2}{2\sigma^2} - \langle \mathbf{x}-\boldsymbol{\mu}, \frac{\boldsymbol{\mu}}{\sigma^2}\rangle \\ &= \frac{\|\mathbf{x}-\boldsymbol{\mu}\|^2}{2\sigma^2} \ . \end{aligned}$$

The function $b_\phi(\mathbf{x})$ in Theorem 5 is given by

$$b_\phi(\mathbf{x}) = \exp(\phi(\mathbf{x}))p_0(\mathbf{x}) = \exp\left(\frac{\|\mathbf{x}\|^2}{2\sigma^2} - \frac{\|\mathbf{x}\|^2}{2\sigma^2}\right)\frac{1}{\sqrt{(2\pi\sigma^2)^d}} = \frac{1}{\sqrt{(2\pi\sigma^2)^d}}\ ,$$

and turns out to be a constant. Thus, $p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}) = \exp(-d_\phi(\mathbf{x},\boldsymbol{\mu}))b_\phi(\mathbf{x})$. qed

**Example 10** Another exponential family that is widely used is the family of multinomial distributions:

$$p(\mathbf{x};\mathbf{q}) = \frac{N!}{\prod_{j=1}^d x_j!}\prod_{j=1}^d q_j^{x_j},$$

where $x_j \in \mathbb{Z}_+$ are frequencies of events, $\sum_{j=1}^d x_j = N$ and $q_j \geq 0$ are probabilities of events, $\sum_{j=1}^d q_j = 1$. As shown below, $p(\mathbf{x};\mathbf{q})$ can be expressed as the density of an exponential distribution in $\mathbf{x} = \{x_j\}_{j=1}^{d-1}$ with natural parameter $\boldsymbol{\theta} = \{\log(\frac{q_j}{q_d})\}_{j=1}^{d-1}$ and cumulant function $\psi(\boldsymbol{\theta}) = -N\log q_d = N\log(1+\sum_{j=1}^{d-1} e^{\theta_j})$.

$$\begin{aligned}
p(\mathbf{x};\mathbf{q}) &= \frac{N!}{\prod_{j=1}^d x_j!}\prod_{j=1}^d q_j^{x_j} \\
&= \exp(\sum_{j=1}^d x_j \log q_j)\frac{N!}{\prod_{j=1}^d x_j!} \ = \ \exp(\sum_{j=1}^{d-1} x_j \log q_j + x_d \log q_d)p_0(\mathbf{x}) \\
&= \exp(\sum_{j=1}^{d-1} x_j \log q_j + (N - \sum_{j=1}^{d-1} x_j)\log q_d)p_0(\mathbf{x}) \\
&= \exp(\sum_{j=1}^{d-1} x_j \log(\frac{q_j}{q_d}) + N\log q_d)p_0(\mathbf{x}) \\
&= \exp(\langle \mathbf{x},\boldsymbol{\theta}\rangle + N\log q_d)p_0(\mathbf{x}) \ = \ \exp(\langle \mathbf{x},\boldsymbol{\theta}\rangle - N\log(\sum_{j=1}^d \frac{q_j}{q_d}))p_0(\mathbf{x}) \\
&= \exp(\langle \mathbf{x},\boldsymbol{\theta}\rangle - N\log(1+\sum_{j=1}^{d-1} e^{\theta_j}))p_0(\mathbf{x}) \ = \ \exp(\langle \mathbf{x},\boldsymbol{\theta}\rangle - \psi(\boldsymbol{\theta}))p_0(\mathbf{x}),
\end{aligned}$$

where $p_0(\mathbf{x})$ is independent of $\boldsymbol{\theta}$. The expectation parameter $\boldsymbol{\mu}$ is given by

$$\boldsymbol{\mu} = \nabla\psi(\boldsymbol{\theta}) = \nabla(N\log(1+\sum_{j=1}^{d-1} e^{\theta_j})) = \left[\frac{Ne^{\theta_j}}{1+\sum_{j=1}^{d-1} e^{\theta_j}}\right]_{j=1}^{d-1} = [Nq_j]_{j=1}^{d-1}$$

and the Legendre conjugate function $\phi$ of $\psi$ is

$$\begin{aligned}\phi(\boldsymbol{\mu}) &= \langle\boldsymbol{\mu},\boldsymbol{\theta}\rangle - \psi(\boldsymbol{\theta}) = \sum_{j=1}^{d-1} Nq_j\log(\frac{q_j}{q_d}) + N\log q_d \\ &= \sum_{j=1}^{d} Nq_j\log q_j = N\sum_{j=1}^{d}(\frac{\mu_j}{N})\log(\frac{\mu_j}{N}),\end{aligned}$$

where $\mu_d = Nq_d$ so that $\sum_{i=1}^{d}\mu_j = N$. Note that $\phi(\boldsymbol{\mu})$ is a constant multiple of negative entropy for the discrete probability distribution given by $\{\frac{\mu_j}{N}\}_{j=1}^{d}$. From Example 2, we know that the corresponding Bregman divergence will be a similar multiple of KL-divergence.

$$\begin{aligned}d_\phi(\mathbf{x},\boldsymbol{\mu}) &= \phi(\mathbf{x}) - \phi(\boldsymbol{\mu}) - \langle\mathbf{x}-\boldsymbol{\mu}, \nabla\phi(\boldsymbol{\mu})\rangle \\ &= N\sum_{j=1}^{d}\frac{x_j}{N}\log(\frac{x_j}{N}) - N\sum_{j=1}^{d}\frac{\mu_j}{N}\log(\frac{\mu_j}{N}) - \sum_{j=1}^{d}(x_j-\mu_j)(1+\log(\frac{\mu_j}{N})) \\ &= N\sum_{j=1}^{d}\frac{x_j}{N}\log(\frac{x_j/N}{\mu_j/N}).\end{aligned}$$

The function $b_\phi(\mathbf{x})$ for this case is given by

$$b_\phi(\mathbf{x}) = \exp(\phi(\mathbf{x}))p_0(\mathbf{x}) = \exp(\sum_{j=1}^{d} x_j\log(\frac{x_j}{N}))\frac{N!}{\prod_{j=1}^{d}x_j!} = \frac{\prod_{j=1}^{d}x_j^{x_j}}{N^N}\frac{N!}{\prod_{j=1}^{d}x_j!}\ ,$$

and $p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}) = \exp(-d_\phi(\mathbf{x},\boldsymbol{\mu}))b_\phi(\mathbf{x})$. ■

## 4.2 Bregman Soft Clustering

Using the correspondence between regular exponential families and regular Bregman divergences, we now pose the Bregman soft clustering problem as a parameter estimation problem for mixture models based on regular exponential family distributions. We revisit the expectation maximization (EM) framework for estimating mixture densities and develop a Bregman soft clustering algorithm (Algorithm 3) for regular Bregman divergences. We also present the Bregman soft clustering algorithm for a set of data points with non-uniform non-negative weights (or measure). We show that the hard clustering algorithm can be interpreted as a special case of the soft clustering algorithm. Further, we discuss an alternative formulation of Bregman clustering in terms of the divergence derived from the conjugate function.

### 4.2.1 Soft Clustering as Mixture Density Estimation

Given a set $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^d$ drawn independently from a stochastic source, consider the problem of modeling the source using a single parametric exponential family distribution. This is the problem of maximum likelihood estimation, or, equivalently, minimum negative log-likelihood estimation of the parameter(s) of a given exponential family distribution. From Theorem 5, minimizing the negative log-likelihood is the same as minimizing the corresponding expected Bregman divergence. Using Proposition 1, we conclude that the optimal distribution is the one with $\boldsymbol{\mu} = \mathbf{E}_\nu[X]$ as the expectation parameter where $X$ is a random variable that takes values in $\mathcal{X}$ following $\nu$ which, by the independence assumption, is the empirical distribution over $\mathcal{X}$. Further, note that the minimum negative log-likelihood of $\mathcal{X}$ under a particular exponential model with log-partition function $\psi$ is the Bregman information of $X$, i.e., $I_\phi(X)$, up to additive constants, where $\phi$ is the conjugate of $\psi$.

Now, consider the problem of modeling the stochastic source with a mixture

of $k$ densities of the same exponential family. The model yields a soft clustering where clusters correspond to the components of the mixture model, and the soft membership of a data point in each cluster is proportional to the probability of the data point being generated by the corresponding density function. For regular Bregman divergences, the *Bregman soft clustering problem* is that of learning the maximum likelihood parameters $\Gamma = \{\boldsymbol{\theta}_h, \pi_h\}_{h=1}^k \equiv \{\boldsymbol{\mu}_h, \pi_h\}_{h=1}^k$ of a mixture model of the form

$$p(\mathbf{x}|\Gamma) = \sum_{h=1}^{k} \pi_h p_{(\psi,\boldsymbol{\theta}_h)}(\mathbf{x}) = \sum_{h=1}^{k} \pi_h \exp(-d_\phi(\mathbf{x}, \boldsymbol{\mu}_h)) b_\phi(\mathbf{x}), \tag{4.13}$$

where the last equality follows from Theorem 5. Since the mixture components are all assumed to be from the same family, the above problem is a special case of the general maximum likelihood parameter estimation problem for mixture models and can be solved by applying the EM algorithm. Note that, by the correspondence between regular Bregman divergences and regular exponential families, (4.13) encompasses the soft clustering problem for *all* regular exponential families.

### 4.2.2 EM for Mixture Models based on Bregman Divergences

Algorithm 2 describes the well known application of EM for mixture density estimation. This algorithm has the property that the likelihood of the data, $L_{\mathcal{X}}(\Gamma)$ is non-decreasing at each iteration. Further, if there exists at least one local maximum for the likelihood function, then the algorithm will converge to a local maximum of the likelihood. For more details, the reader is referred to [Col97, MK96] and [Bil97].

The Bregman soft clustering problem is to estimate the maximum likelihood parameters for the mixture model given in (4.13). Using the Bregman divergence viewpoint, we get a simplified version of the above EM algorithm that we call the Bregman soft clustering algorithm (Algorithm 3). Using Proposition 1, the computa-

**Algorithm 2** Standard EM for Mixture Density Estimation

**Input:** Set $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subseteq \mathbb{R}^d$, number of clusters $k$.

**Output:** $\Gamma^\dagger$: local maximizer of $L_\mathcal{X}(\Gamma) = \prod_{i=1}^n (\sum_{h=1}^k \pi_h p_h(\mathbf{x}_i|\boldsymbol{\theta}_h))$ where $\Gamma = \{\boldsymbol{\theta}_h, \pi_h\}_{h=1}^k$, soft partitioning $\{\{p(h|\mathbf{x}_i)\}_{h=1}^k\}_{i=1}^n$.

**Method:**

Initialize $\{\boldsymbol{\theta}_h, \pi_h\}_{h=1}^k$ with some $\boldsymbol{\theta}_h \in \Theta$, and $\pi_h \geq 0$, $\sum_{h=1}^k \pi_h = 1$

**repeat**

{The Expectation Step}

**for** $i = 1$ to $n$ **do**

**for** $h = 1$ to $k$ **do**

$p(h|\mathbf{x}_i) \leftarrow \frac{\pi_h p_h(\mathbf{x}_i|\boldsymbol{\theta}_h)}{\sum_{h'=1}^k \pi_{h'} p_{h'}(\mathbf{x}_i|\boldsymbol{\theta}_{h'})}$

**end for**

**end for**

{The Maximization Step}

**for** $h = 1$ to $k$ **do**

$\pi_h \leftarrow \frac{1}{n} \sum_{i=1}^n p(h|\mathbf{x}_i)$

$\boldsymbol{\theta}_h \leftarrow \operatorname{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^n \log(p_h(\mathbf{x}_i|\boldsymbol{\theta}))p(h|\mathbf{x}_i)$

**end for**

**until** *convergence*

return $\Gamma^\dagger = \{\boldsymbol{\theta}_h, \pi_h\}_{h=1}^k$

**Algorithm 3** Bregman Soft Clustering

**Input:** Set $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{S} \subseteq \mathbb{R}^d$, Bregman divergence $d_\phi : \mathcal{S} \times \text{int}(\mathcal{S}) \mapsto \mathbb{R}$, number of clusters $k$.

**Output:** $\Gamma^\dagger$, local maximizer of $\prod_{i=1}^n (\sum_{h=1}^k \pi_h f_\phi(\mathbf{x}_i) \exp(-d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h)))$ where $\Gamma = \{\boldsymbol{\mu}_h, \pi_h\}_{h=1}^k$, soft partitioning $\{\{p(h|\mathbf{x}_i)\}_{h=1}^k\}_{i=1}^n$

**Method:**

Initialize $\{\boldsymbol{\mu}_h, \pi_h\}_{h=1}^k$ with some $\boldsymbol{\mu}_h \in \text{int}(\mathcal{S}), \pi_h \geq 0$, and $\sum_{h=1}^k \pi_h = 1$

**repeat**

{The Expectation Step}

**for** $i = 1$ to $n$ **do**

**for** $h = 1$ to $k$ **do**

$p(h|\mathbf{x}_i) \leftarrow \frac{\pi_h \exp(-d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_h))}{\sum_{h'=1}^k \pi_{h'} \exp(-d_\phi(\mathbf{x}_i, \boldsymbol{\mu}_{h'}))}$

**end for**

**end for**

{The Maximization Step}

**for** $h = 1$ to $k$ **do**

$\pi_h \leftarrow \frac{1}{n} \sum_{i=1}^n p(h|\mathbf{x}_i)$

$\boldsymbol{\mu}_h \leftarrow \frac{\sum_{i=1}^n p(h|\mathbf{x}_i)\mathbf{x}_i}{\sum_{i=1}^n p(h|\mathbf{x}_i)}$

**end for**

**until** *convergence*

return $\Gamma^\dagger = \{\boldsymbol{\mu}_h, \pi_h\}_{h=1}^k$

tionally intensive M-step turns out to be very simple to solve. In fact, the Bregman divergence viewpoint gives an alternative interpretation of a well known efficient EM scheme applicable to learning a mixture of exponential distributions [RW84]. The resulting update equations are similar to those for learning mixture models of identity covariance Gaussians. Note that these equations are applicable to mixtures of any regular exponential distributions, as long as $\mathbf{x}$ is the (minimal) sufficient statistic vector.

It is important to note that the simplification of the M-step is applicable only when the parameterization is with respect to the expectation parameter space, i.e., when $d_\phi$ corresponding to an exponential family is known. Otherwise, if the parameterization is with respect to the natural parameter space, i.e., the functional form for a family is known in terms of its cumulant $\psi$ and natural parameters $\boldsymbol{\theta}$, to obtain $\phi(\boldsymbol{\mu}) = \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle - \psi(\boldsymbol{\theta})$, the problem

$$\boldsymbol{\mu}(\boldsymbol{\theta}) = \operatorname*{argsup}_{\mathbf{x} \in \mathbb{R}^d} \left( \langle \boldsymbol{\theta}, \mathbf{x} \rangle - \psi(\boldsymbol{\theta}) \right) , \tag{4.14}$$

needs to be solved. Since the function to be maximized in (4.14) is precisely the log-likelihood of the exponential family density (with respect to an appropriate measure), the transformation is equivalent to solving a maximum likelihood estimation problem, which is computationally expensive for several exponential family distributions. In such a situation, transforming the problem to the expectation space need not lead to any tangible computational benefits. However, if the Bregman divergence $d_\phi$ corresponding to an exponential family is either known or easy to compute from the natural parameterization, then Algorithm 3 is computationally much more efficient. In fact, in some situations it may be easier to design regular Bregman divergences for mixture modeling of data than to come up with an appropriate exponential family. Such situations can take full advantage of the computationally efficient Bregman soft clustering algorithm.

Finally we show that Algorithms 2 and 3 are exactly equivalent. the Bregman soft The following result shows how Proposition 1 and Theorem 5 can be used to simplify the M-step of Algorithm 2. Note that Proposition 4 has appeared in various forms in the literature (see, for example, [RW84, MK96]). We give an alternative proof using Bregman divergences.

**Proposition 4** *For a mixture model with density given by (4.13), the maximization step for the density parameters in the EM algorithm (Algorithm 2), $\forall h, 1 \leq h \leq k$, reduces to:*

$$\boldsymbol{\mu}_h = \frac{\sum_{i=1}^{n} p(h|\mathbf{x}_i)\mathbf{x}_i}{\sum_{i=1}^{n} p(h|\mathbf{x}_i)}. \tag{4.15}$$

*Proof:* The maximization step for the density parameters in the EM algorithm, $\forall h, 1 \leq h \leq k$, is given by

$$\boldsymbol{\theta}_h = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log(p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}_i))p(h|\mathbf{x}_i).$$

For the given mixture density, the component densities, $\forall h, 1 \leq h \leq k$, are given by

$$p_{(\psi,\boldsymbol{\theta}_h)}(\mathbf{x}) = b_\phi(\mathbf{x}) \exp(-d_\phi(\mathbf{x}, \boldsymbol{\mu}_h)).$$

Substituting the above into the maximization step, we obtain the update equations for the expectation parameters $\boldsymbol{\mu}_h$, $1 \leq h \leq k$,

$$\begin{aligned}
\boldsymbol{\mu}_h &= \operatorname*{argmax}_{\boldsymbol{\mu}} \sum_{i=1}^{n} \log(b_\phi(\mathbf{x}_i) \exp(-d_\phi(\mathbf{x}_i, \boldsymbol{\mu})))p(h|\mathbf{x}_i) \\
&= \operatorname*{argmax}_{\boldsymbol{\mu}} \sum_{i=1}^{n} (\log(b_\phi(\mathbf{x}_i)) - d_\phi(\mathbf{x}_i, \boldsymbol{\mu}))p(h|\mathbf{x}_i) \\
&= \operatorname*{argmin}_{\boldsymbol{\mu}} \sum_{i=1}^{n} d_\phi(\mathbf{x}_i, \boldsymbol{\mu})p(h|\mathbf{x}_i) \text{ (as } b_\phi(\mathbf{x}) \text{ is independent of } \boldsymbol{\mu})
\end{aligned}$$

$$= \operatorname*{argmin}_{\boldsymbol{\mu}} \sum_{i=1}^{n} d_\phi(\mathbf{x}_i, \boldsymbol{\mu}) \frac{p(h|\mathbf{x}_i)}{\sum_{i'=1}^{n} p(h|\mathbf{x}_{i'})}.$$

From Proposition 1, we know that the expected Bregman divergence is minimized by the expectation of $\mathbf{x}$, i.e.,

$$\operatorname*{argmin}_{\boldsymbol{\mu}} \sum_{i=1}^{n} d_\phi(\mathbf{x}_i, \boldsymbol{\mu})) \frac{p(h|\mathbf{x}_i)}{\sum_{i'=1}^{n} p(h|\mathbf{x}_{i'})} = \frac{\sum_{i=1}^{n} p(h|\mathbf{x}_i)\ \mathbf{x}_i}{\sum_{i=1}^{n} p(h|\mathbf{x}_i)}.$$

Therefore, the update equation for the parameters is just a weighted averaging step,

$$\boldsymbol{\mu}_h = \frac{\sum_{i=1}^{n} p(h|\mathbf{x}_i)\mathbf{x}_i}{\sum_{i=1}^{n} p(h|\mathbf{x}_i)}, \qquad \forall h, 1 \leq h \leq k.$$

That completes the proof. ∎

The update equations for the posterior probabilities (E-step) $\forall \mathbf{x} \in \mathcal{X},\ \forall h, 1 \leq h \leq k$, are given by

$$p(h|\mathbf{x}) = \frac{\pi_h \exp(-d_\phi(\mathbf{x}, \boldsymbol{\mu}_h))}{\sum_{h'=1}^{k} \pi_{h'} \exp(-d_\phi(\mathbf{x}, \boldsymbol{\mu}_{h'}))}$$

as the $b_\phi(\mathbf{x})$ factor cancels out. The prior update equations are independent of the parametric form of the densities and remain unaltered. Hence, for a mixture model with density given by (4.13), the EM algorithm (Algorithm 2) reduces to the Bregman soft clustering algorithm (Algorithm 3).

Thus far we have considered the Bregman soft clustering problem for a set $\mathcal{X}$ where all the elements are equally important and assumed to have been independently sampled from some particular exponential distribution. In practice, it might be desirable to associate weights $\nu_i$ with the individual samples such that $\sum_i \nu_i = 1$ and optimize a weighted log-likelihood function. A slight modification to the M-step of the Bregman soft clustering algorithm is sufficient to address this new optimization problem. The E-step remains identical and the new update equations

for the M-step $\forall h, 1 \leq h \leq k$, are given by

$$\begin{aligned} \pi_h &= \sum_{i=1}^{n} \nu_i p(h|\mathbf{x}_i), \\ \boldsymbol{\mu}_h &= \frac{\sum_{i=1}^{n} \nu_i p(h|\mathbf{x}_i)\mathbf{x}_i}{\sum_{i=1}^{n} \nu_i p(h|\mathbf{x}_i)}. \end{aligned}$$

Finally, we note that the Bregman hard clustering algorithm is a limiting case of the above soft clustering algorithm. For every convex function $\phi$ and positive constant $\beta$, $\beta\phi$ is also a convex function with the corresponding Bregman divergence $d_{\beta\phi} = \beta d_\phi$. In the limit, when $\beta \to \infty$, both the E and M steps of the soft clustering algorithm reduce to the assignment and re-estimation step of the hard clustering algorithm.

### 4.2.3 An Alternative Formulation for Bregman Clustering

In earlier sections, Bregman divergence was measured with the data points as the first argument and the cluster representative as the second argument. Since Bregman divergences are not symmetric (with the exception of squared Euclidean distance), we now consider an alternative formulation of Bregman clustering where cluster representatives are the first argument of the Bregman divergence. Using Legendre duality, we show that this alternate formulation is equivalent to our original Bregman clustering problem in a dual space using a different, but uniquely determined Bregman divergence.

We focus on the hard clustering case. Let $X$ be a random variable that takes values in $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ following a positive probability measure $\nu$. Then the alternative Bregman hard clustering problem is to find clusters $\{\mathcal{X}_h\}_{h=1}^k$ and corresponding representatives $\{\boldsymbol{\mu}_h\}_{h=1}^k$ that solve

$$\min_{\{\mu_h\}_{h=1}^k} \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \nu_i d_\phi(\boldsymbol{\mu}_h, \mathbf{x}_i). \tag{4.16}$$

As mentioned earlier, Bregman divergences are convex in the first argument and hence, the resulting optimization problem for each cluster is convex so there is a unique optimal representative for each cluster. However, unlike in the original formulation, the optimal cluster representative is not always the expectation and depends on the Bregman divergence $d_\phi$.

It is interesting to note that this alternative formulation, though seemingly different, reduces to the original formulation with an appropriate representation. Let $\phi$ be the generating convex function of $d_\phi$ such that $(\text{int}(\text{dom}(\phi)), \phi)$ is a convex function of Legendre type and let $(\text{int}(\text{dom}(\psi)), \psi)$ be the corresponding Legendre dual. Then for any $\mathbf{x}, \mathbf{y} \in \text{int}(\text{dom}(\phi))$, the Bregman divergence $d_\phi(\mathbf{x}, \mathbf{y}) = d_\psi(\boldsymbol{\theta}_\mathbf{y}, \boldsymbol{\theta}_\mathbf{x})$ where $d_\psi$ is the Bregman divergence derived from $\psi$ and $\boldsymbol{\theta}_\mathbf{x} = \nabla\phi(\mathbf{x}), \boldsymbol{\theta}_\mathbf{y} = \nabla\phi(\mathbf{y})$ (Appendix A, Property 6). Using the above property, we can restate the alternative Bregman clustering problem in the dual space. More specifically, let $\mathcal{X}^{\boldsymbol{\theta}} = \{\boldsymbol{\theta}_{\mathbf{x}_i}\}_{i=1}^n$ where $\boldsymbol{\theta}_{\mathbf{x}_i} = \nabla\phi(\mathbf{x}_i),\ \forall \mathbf{x}_i, 1 \leq i \leq n$, and let $\boldsymbol{\theta}_h = \nabla\phi(\boldsymbol{\mu}_h),\ \forall \boldsymbol{\mu}_h, 1 \leq h \leq k$. Then, the alternative Bregman hard clustering problem can be expressed as

$$\min_{\{\boldsymbol{\theta}_h\}_{h=1}^k} \sum_{h=1}^{k} \sum_{\boldsymbol{\theta}_{\mathbf{x}_i} \in \mathcal{X}_h^{\boldsymbol{\theta}}} \nu_i d_\psi(\boldsymbol{\theta}_{\mathbf{x}_i}, \boldsymbol{\theta}_h). \tag{4.17}$$

where $\mathcal{X}_h^{\boldsymbol{\theta}}$ correspond to cluster $h$ in the dual space. It is now straightforward to see that this is our original Bregman hard clustering problem for the set $\mathcal{X}^{\boldsymbol{\theta}}$ consisting of the dual data points with the same measure $\nu$ and the dual Bregman divergence $d_\psi$. The optimal cluster representative in this dual space is given by the expectation, which is easy to compute. The efficiency of this approach has the same premise as the efficient EM scheme for exponential families, i.e, the M-step can be simplified if there is an easy way to transition to the dual space.

# Chapter 5

# Clustering on the Hypersphere

## 5.1 Motivation

In some applications, the data points to be clustered reside in a high-dimensional feature space even after suitable pre-processing, including feature selection, have been carried out. For example, in the popular vector space representation of text documents, the dimensionality of the feature space is the size of the vocabulary [BYRN99]. Even after stemming is carried out and both very rare as well as very common terms have been discarded, the residual vocabulary often contains thousands of terms. In market basket analysis of large retail data, the number of significant products, each represented by one feature, may run in the thousands [SG02]. Both text and market basket data also have other properties in common: the data matrix is typically very sparse and only contains non-negative entries, and the underlying clusters are highly non-Gaussian in nature.

In general, sparsity and other issues due to the "curse of dimensionality" [Fri94] make the clustering of highly non-Gaussian, high-dimensional data a very difficult problem to solve using density based approaches. Partitional methods such

[0]The work presented in this chapter has earlier appeared as [BG04].

as $k$-means and its variants also fail miserably since the underlying assumption that the data can be well modeled by a mixture of $k$ Gaussians (value of $k$ being prespecified) with identical covariance matrices, is violated. The way out is to exploit domain knowledge about the properties of the data and of the desired clusters. For example, if it is known that the data actually resides in a manifold that is of much lower dimensions than the embedding space, solving for a mixture of principal surfaces can help [CG01, HDR97].

This chapter is aimed at applications for which the domain knowledge indicates that data is directional [Mar75]. For such scenarios, the curse of dimensionality is somewhat alleviated by normalizing the data to unit $L_2$ norm, since only the direction of a data vector is relevant. For example, many methods for document clustering normalize the document vectors to unit length after all other preprocessing and normalizations such as TF-IDF have been carried out. The cosine of the angle between two such normalized vectors then serves as the default similarity measure between the two documents that they represent. Normalization prevents larger documents from dominating the clustering results, and can be viewed as an application of domain-specific characteristics and requirements to alleviate the "curse of dimensionality" problems. A noteworthy algorithm for clustering normalized document vectors is spherical kmeans (`spkmeans`) [DM01, DFG01], in which the cluster representatives are also constrained to be of unit length, allocation of data points to their nearest representatives is based on cosine similarity, and, after a full pass through the data, the updated locations of the representatives are based on maximizing the average cosine similarity between the cluster "center" and all the points assigned to that cluster. Note that, like `kmeans`, `spkmeans` is also a batch-iterative procedure. This algorithm has been successfully used to cluster text documents in 2000+ dimensional space, providing superior cluster definitions in the process [DM01, DFG01]. We shall show in Section 2 that `spkmeans` is really a batch version

of a competitive learning algorithm.

Normalization of high-dimensional vectors before clustering is also fruitful for some other applications. In fact, it is meaningful for market basket data analysis if one is interested in, say, grouping customers based on the similarities between the percentages of their money spent on the various products.

Having shown the need for clustering high dimensional data residing on hyperspheres by drawing examples from document clustering and market basket analysis, we use the same two domains to motivate the desirability of obtaining *balanced* clusters, i.e., clusters of comparable sizes [BG02b, BBD00]. In general, the natural clusters in the data may be of widely varying sizes, this variation may not be known beforehand and balanced solutions may not be important. However, as discussed in section 2.2, several applications in market basket analysis and text clustering demand comparably sized segmentations of the data. In addition to application requirements, balanced clustering is sometimes also helpful because it tends to decrease sensitivity to initialization and to avoid outlier clusters (highly under-utilized representatives) from forming, and thus has a beneficial regularizing effect even in situations where balancing is not a requirement. This will be evident from our experimental results in Section 5.6.

Unfortunately, `KMeans` type algorithms (including the EM approach) as well as the basic on-line competitive learning mechanisms for clustering are increasingly prone to yielding imbalanced solutions as the input dimensionality increases. This problem is exacerbated when a large (tens or more) number of clusters are needed, and it is well known that both hard and soft `kmeans` invariably result in some near-empty clusters in such scenarios.

**Competitive Learning:** Competitive learning techniques employ winner-take-all mechanisms to determine the most responsive cell to a given input [Gro76, RZ85, Gro87]. If this cell or exemplar then adjusts its afferent weights to respond even

more strongly to the given input, the resultant system can be shown to perform unsupervised clustering. For example, the non-normalized competitive learning version of Rumelhart and Zipser [RZ85], essentially yields an on-line analogue of the popular `k-means` clustering algorithm. There are also soft competitive learning methods with multiple winners per input [ZL02], that can be viewed as on-line analogues of soft batch-iterative clustering algorithms such as fuzzy c-means [BP92] as well as the expectation-maximization (EM)-based approach to clustering data modeled as a mixture of Gaussians [Bli98].

To address the problem of obtaining clusters of widely varying sizes, a "conscience" mechanism was proposed for competitive learning in 1988 [deS88], that made frequently winning representatives less likely to win in the future because of their heavier conscience. This work was followed by the notable frequency sensitive competitive learning(FSCL) method [AKCM90]. FSCL was originally formulated to remedy the problem of under-utilization of parts of a codebook in vector quantization. Motivated by earlier work of Grossberg [Gro76], the conscience mechanism used in FSCL multiplicatively scaled the distortion (distance of the exemplar or codebook vector from the input) by the number of times that exemplar was the winner in the past. Thus highly winning exemplars were discouraged from attracting new inputs. However, this mechanism was not derived from first principles or applied to high-dimensional, normalized clustering.

## 5.2 Clustering on a Hypersphere

The classical `kmeans` clustering algorithm gives the (local) maximum likelihood estimates of the means of $k$ Gaussians [Mit97] under certain assumptions [KMN97, BBM02] by using the Expectation Maximization (EM) algorithm [DLR77]. The EM algorithm is guaranteed to give a local optimum for these maximum likelihood estimates. We use a similar approach for deriving the `spkmeans` algorithm for clustering

points on the surface of a hypersphere. The von Mises-Fisher (vMF) distribution is an analogue of the Gaussian distribution on a hypersphere [Mar75, SK01, BDGS03] in that it is the maximum entropy distribution on the hypersphere when the first moment is fixed [KK92] under the constraint that the points are on a unit hypersphere. The density of a $d$-dimensional vMF distribution is given by

$$f(\mathbf{x}; \boldsymbol{\mu}, \kappa) = \frac{1}{Z_d(\kappa)} \exp\left(\kappa \mathbf{x}^T \boldsymbol{\mu}\right), \tag{5.1}$$

where $\boldsymbol{\mu}$, with $\|\boldsymbol{\mu}\| = 1$, represents the mean direction vector and $\kappa$ is the dispersion around the mean, analogous to the mean and covariance for the multivariate Gaussian distribution. The normalizing coefficient is

$$Z_d(\kappa) = (2\pi)^{d/2} I_{d/2-1}(\kappa) / \kappa^{d/2-1}, \tag{5.2}$$

where $I_r(y)$ is the modified Bessel function of the first kind and order $r$ [McL55]. Assume that there are $n$ data points $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ on the surface of a unit hypersphere and there are $k$ vMF distributions $f_h, h = 1, \cdots, k$, such that each point has been generated following exactly one of these distributions. We want to estimate the parameters of the $k$ vMF distributions so that the likelihood of the observed data is maximized. Like the Euclidean `kmeans` case, we initially assume $\kappa$ to be a constant for all the $k$ distributions. Let $\mathcal{Z} = \{\mathrm{z}_1, \ldots, \mathrm{z}_n\}$ be the set of hidden random variables over $\{1, \ldots, k\}$ corresponding to the set of data points $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ so that $\mathrm{z}_i = h$ if $\mathbf{x}_i$ was generated following $f_h$ and 0 otherwise. Assuming the data points have been drawn independently, the log-likelihood of the observed data is given by

$$\mathcal{L}(\mathcal{X}|\Gamma) = \sum_{i=1}^{n} \ln f(\mathbf{x}_i; \boldsymbol{\mu}_{\mathrm{z}_i}, \kappa) = \sum_{i=1}^{n} (\kappa \mathbf{x}_i^T \boldsymbol{\mu}_{\mathrm{z}_i} - \ln(Z_d(\kappa))), \tag{5.3}$$

where $\boldsymbol{\mu}_{z_i}$ is the mean of the vMF distribution that generated $\mathbf{x}_i$. Since $\kappa$ is a constant, maximizing this log-likelihood with respect to the $\boldsymbol{\mu}_h, h = 1, \ldots, k$, is same as maximizing

$$\mathcal{J} = \sum_{i=1}^{n} \mathbf{x}_i^T \boldsymbol{\mu}_{z_i} \tag{5.4}$$

under the constraint that $\|\mu_h\| = 1, \forall h$. Since the parameters $\mu_h$ as well as the distributions of the random variables $z_i$ are not known, we can use the EM algorithm to do maximum likelihood estimation under incomplete information. If the parameters are set to random values in a primal M-step, the E-step of the algorithm involves *assigning* the data points to the most likely vMF distribution to have generated it. More precisely, we compute $h^*$ so that[1]

$$h^* = \operatorname*{argmax}_h \ \ln f(\mathbf{x}_i; \boldsymbol{\mu}_h) = \operatorname*{argmax}_h \ \mathbf{x}_i^T \boldsymbol{\mu}_h. \tag{5.5}$$

Then, the M-step involves computing the $\mu_h, h = 1, \cdots, k$, using the current assignments of the data. The parameters are computed by maximizing the expected log-likelihood, the expectation being over the distribution of the $z_i$s. Note that unlike many other applications of the M-step, the maximization in this case is a constrained maximization with the constraints $\|\mu_h\| = 1, \forall h$ and hence is done by using the Lagrange multiplier method. Let $\lambda_h$ be the Lagrange multiplier corresponding to the constraint $\boldsymbol{\mu}_h^T \boldsymbol{\mu}_h = 1$.[2] Then the Lagrangian is given by

$$\begin{aligned} L(\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_k, \lambda_1, \cdots, \lambda_k; \mathcal{X}) &= \sum_{i=1}^{n} \mathbf{x}_i^T \boldsymbol{\mu}_h + \sum_{h=1}^{k} \lambda_h (\boldsymbol{\mu}_h^T \boldsymbol{\mu}_h - 1) \\ &= \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \mathbf{x}_i^T \boldsymbol{\mu}_h + \sum_{h=1}^{k} \lambda_h (\boldsymbol{\mu}_h^T \boldsymbol{\mu}_h - 1), \end{aligned}$$

[1] One can also formulate a soft assignment based algorithm based on vMF distributions, but soft clustering is outside the scope of this chapter. See [BDGS03] for details.

[2] There is a subtle difference between the constraints $\|\boldsymbol{\mu}_h\| = 1$ and $\boldsymbol{\mu}_h^T \boldsymbol{\mu}_h = 1$. Since this difference is not important in the present analysis, we choose to ignore it for simplicity.

where $\mathcal{X}_h$ is a set such that if the current $z_i = h$, then $\mathbf{x}_i \in \mathcal{X}_h$. In other words, $\mathcal{X}_h$ the set of points assigned to the current $h$-th cluster. Now, taking derivatives of the Lagrangian with respect to $\boldsymbol{\mu}_h$ and $\lambda_h$, and setting it to zero, for $h = 1, \cdots, k$, we get the following equations:

$$\boldsymbol{\mu}_h = \frac{1}{2\lambda_h} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \mathbf{x}_i, \tag{5.6}$$

$$\boldsymbol{\mu}_h^T \boldsymbol{\mu}_h = 1. \tag{5.7}$$

Substituting (5.6) into (5.7), we get

$$\lambda_h = \frac{1}{2} \parallel \sum_{\mathbf{x}_i \in \mathcal{X}_h} \mathbf{x}_i \parallel. \tag{5.8}$$

Now, replacing $\lambda_h$ in (5.6) with (5.8), we get the final M-step as

$$\boldsymbol{\mu}_h = \frac{\sum_{\mathbf{x}_i \in \mathcal{X}_h} \mathbf{x}_i}{\parallel \sum_{\mathbf{x}_i \in \mathcal{X}_h} \mathbf{x}_i \parallel}. \tag{5.9}$$

Repeating the steps given in (5.5) and (5.9) results in an iterative relocation scheme that, being an EM algorithm, is guaranteed to give a local maxima of the likelihood function in (5.3) and hence also (5.4) after convergence. This scheme was introduced as the spherical kmeans (`spkmeans`) algorithm by Dhillon *et. al.* [DM01] since the data points lie on the surface of the unit hypersphere. We have now provided a derivation of the same from maximum likelihood principles. In fact, we have obtained a batch mode version of normalized competitive learning [RZ85]. Note that while the performance of this algorithm can be evaluated using (5.3), the following objective function obtained by adding constant additive and multiplicative factors,

is simpler and more interpretable:

$$\bar{\mathcal{J}} = \frac{1}{n} \sum_{h=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{X}_h} \mathbf{x}_i^T \boldsymbol{\mu}_h. \tag{5.10}$$

$\bar{\mathcal{J}}$ can be interpreted as the average *cosine similarity* (cosine of the angle) between any vector $\mathbf{x}_i$ and its cluster representative $\boldsymbol{\mu}_{z_i}$. It serves as an intrinsic measure of cluster quality and will be called the `spkmeans` objective function.

## 5.3 Frequency Sensitive Assignments

From empirical studies [DM01, BG02a, BDGS03] in clustering, `spkmeans` has been shown to be clearly superior to regular `kmeans` for directional data [BDGS03]. However, like its Euclidean space counterpart, it quite often gets stuck in poor local solutions resulting in empty clusters or clusters having very few points, for moderately large values of $k$. Both the formulations do not have any explicit way to guard against such a scenario. A similar problem had been reported in the signal processing community for the problem of vector quantization where some parts of the codebook were under-utilized as a result of poor local solutions to the optimization problem for codebook generation [RZ85]. The problem was empirically addressed by using frequency sensitive competitive learning(FSCL) [AKCM90, GA96]. FSCL is a conscience type competitive learning approach that overcomes the problems associated with simple competitive learning [AKCM90] and Kohonen's self-organizing feature maps in vector quantization applications. In FSCL, the competitive computing units are penalized in proportion to the frequency of their winning, so that eventually all units participate in the quantization of the data space. Convergence properties of the FSCL algorithm to a local minima have been studied by approximating the final phase of the FSCL by a diffusion process described by a Fokker-Plank equation [GMA97].

As mentioned previously, the `kmeans` algorithm can be viewed as an EM algorithm on a mixture of identity variance Gaussians assuming the cluster assignment hidden variables are distributed such that they take one value in $\{1, \cdots, k\}$ (for hard assignments) [BBM02]. A frequency sensitive learning mechanism can be derived from this mixture of Gaussians framework by making each of the Gaussians *shrink* in proportion to the number of points that have been assigned to it. More precisely, if $n_h$ points have been assigned to the $h$-th cluster, the covariance of its representative Gaussian is set to $\Sigma_h = \frac{1}{n_h}\mathbb{I}$ where $\mathbb{I}$ is the identity matrix. Note that if $n_h$ is large for a particular $h$, then that Gaussian shrinks more in the sense that the density gets more peaked around the mean. The log-likelihood of a particular point $\mathbf{x}$ with respect to this Gaussian is given by

$$\begin{aligned}\mathcal{L}(\mu_h, n_h|\mathbf{x}) &= \ln \frac{1}{\sqrt{(2\pi)^d|\Sigma_h|}} \exp(-\frac{1}{2}(\mathbf{x}-\mu_h)^T\Sigma_h^{-1}(\mathbf{x}-\mu_h)) \\ &= -\frac{n_h}{2}\|\mathbf{x}-\mu_h\|^2 - \frac{d}{2}\ln n_h - \frac{d}{2}\ln 2\pi,\end{aligned}$$

where $d$ is the dimensionality of the data. Since data points are assigned to the most likely Gaussian to have generated it, the point $\mathbf{x}$ will be assigned to the cluster $h^*$ where

$$\begin{aligned}h^* &= \operatorname*{argmax}_h \ \mathcal{L}(\mu_h, n_h|\mathbf{x}) \\ &= \operatorname{argmin}_h \ \left\{n_h\|\mathbf{x}-\mu_h\|^2 + d\ln n_h\right\}. \qquad (5.11)\end{aligned}$$

Thus, the higher $n_h$ is, the lower is the chances of a point getting assigned to that cluster. This is exactly what any FSCL tries to achieve. Interestingly, the empirically proposed FSCL method [AKCM90] only considers $n_h\|x-\mu_h\|^2$. Our formal treatment of the idea results in an extra second term, namely $d\ln n_h$.

Using the same basic idea, we propose a change in the formulation of spherical

kmeans in order to prevent poor local solutions. Rather than keeping $\kappa$ constant, we propose to make it inversely proportional to the number of points assigned to the corresponding distribution. Thus, if $n_h$ is the number of points assigned to $f_h$, then we set $\kappa_h \propto 1/n_h$. Intuitively, this is akin to using shrinking Gaussians in the Euclidean space in the sense that as more points are assigned to a particular cluster, the "width" of its representative Gaussian reduces. As a result, effective distance of points from this cluster increases, or, in the spherical case, the similarity of points from this cluster decreases. Thus, if a point $\mathbf{x}$ is such that $\mathbf{x}^T\boldsymbol{\mu}_1 = \mathbf{x}^T\boldsymbol{\mu}_2$ but $n_{h_1} < n_{h_2}$, then $\mathbf{x}$ has a higher likelihood of having been generated from $f_{h_1}$ than $f_{h_2}$ in the frequency sensitive setting. Hence, the likelihood of points going to clusters having less number of points is higher and this implicitly discourages poor local solutions having empty clusters or clusters having very small number of points.

Formally, let $\kappa_h \propto 1/n_h \Rightarrow \kappa_h = c/n_h$, where $c$ is a suitable proportionality constant. Then, the log-likelihood of data-point $\mathbf{x}_i$ having been generated from $f_h$ is given by

$$\begin{aligned}
\log\ f(\mathbf{x}_i; \boldsymbol{\mu}_h, \kappa_h) &= \frac{c}{n_h}\,\mathbf{x}_i^T\boldsymbol{\mu}_h - \log Z_d\left(\frac{c}{n_h}\right) \\
&\equiv \frac{c}{n_h}\,\mathbf{x}_i^T\boldsymbol{\mu}_h - \log(I_{d/2-1}(c/n_h)) - (\frac{d}{2}-1)\log n_h,
\end{aligned}$$

where $\mathcal{F}(h) \equiv \mathcal{G}(h)$ means that $\text{argmax}_h\ \mathcal{F}(h) = \text{argmax}_h\ \mathcal{G}(h)$. For simplifying the expression further, we choose $c = nd^2/2k$. Then, noting that $n_h = O(n/k)$ and $d$ is a large number so that $nd^2/2kn_h \gg d$, using the fact that $I_n(x) \approx e^x/\sqrt{2\pi x}$ for fixed $n$ and $x \gg n$ [McL55], we get

$$\begin{aligned}
\log\ I_{d/2-1}(c/n_h) &= \log I_{d/2-1}\left(\frac{n/k}{n_h}\cdot\frac{d^2}{2}\right) \\
&\approx \frac{n/k}{n_h}\cdot\frac{d^2}{2} - \frac{1}{2}\log(2\pi\frac{n/k}{n_h}\cdot\frac{d^2}{2})
\end{aligned}$$

$$\equiv \quad \frac{n/k}{n_h} \cdot \frac{d^2}{2} + \frac{1}{2} \log n_h.$$

Hence,

$$\begin{aligned}
\log\ f(\mathbf{x}_i; \boldsymbol{\mu}_h, \kappa_h) &\equiv \frac{n/k}{n_h} \cdot \frac{d^2}{2} \mathbf{x}_i^T \boldsymbol{\mu}_h - \frac{n/k}{n_h} \cdot \frac{d^2}{2} - \frac{1}{2} \log n_h - (\frac{d}{2} - 1) \log n_h \\
&= \frac{n/k}{n_h} \cdot \frac{d^2}{2} \left( \mathbf{x}_i^T \boldsymbol{\mu}_h + 1 \right) - \frac{d-1}{2} \log n_h \\
&\approx \frac{d}{2} \left[ \frac{(n/k)d}{n_h} \left( \mathbf{x}_i^T \boldsymbol{\mu}_h + \boldsymbol{\mu}_h^T \boldsymbol{\mu}_h \right) - \log n_h \right] \\
&\equiv \frac{(n/k)d}{n_h} (\mathbf{x}_i + \boldsymbol{\mu}_h)^T \boldsymbol{\mu}_h - \log n_h.
\end{aligned}$$

Hence, the most likely distribution to have generated the point $\mathbf{x}_i$ is given by

$$h^* = \operatorname*{argmax}_h \left\{ \frac{(n/k)d}{n_h} (\mathbf{x}_i + \boldsymbol{\mu}_h)^T \boldsymbol{\mu}_h - \log n_h \right\} \tag{5.12}$$

$$= \operatorname*{argmax}_h \frac{1}{n_h} \left\{ \mathbf{x}_i^T \boldsymbol{\mu}_h + 1 - \frac{n_h}{(n/k)d} \log n_h \right\}. \tag{5.13}$$

Note, from (5.13), that the spherical kmeans assignment function (5.5) now gets a multiplicative and an additive term, both of which penalize larger clusters. Further note that this particular form of the likelihood function is due to our choice of the proportionality constant and other values of the constant will give slightly different forms for this likelihood function. However, this particular choice helps us use an asymptotic behavior of the modified Bessel function of the first kind thereby making the formulation computationally tractable.

In the next few sections, we shall present algorithms for clustering data based on the frequency sensitive assignment rules as derived above (5.13). We focus on two types of problems: (a) in which the data points are static and the algorithm can read the data as many times as required, and (b) in which the data points are streaming so that algorithm can read every data point exactly once. In section 5.4,

we present three algorithms for clustering static data: (i) `fs-spkmeans` is a direct extension of `spkmeans` using the frequency sensitive assignments from (5.13); (ii) `pifs-spkmeans` is a partially incremental version of `fs-spkmeans` where the effective number of points per cluster are updated incrementally after processing every point and the mean of every cluster is updated in batch once in every iteration (after processing all the points); and (iii) `fifs-spkmeans` is a fully incremental version of `fs-spkmeans` where both the effective number of points per cluster and the cluster means are updated after processing every point. Note that all these algorithms need to know the number of points to be processed up-front and hence are applicable to static data only. In section 5.5, we present `sfs-spkmeans`, an algorithm for frequency sensitive clustering of streaming data, where knowing the number of data points to be clustered is not necessary.

## 5.4 Algorithms for Static Data

Based on the analysis presented in section 5.3 and the frequency sensitive assignments according to (5.13), we first present the algorithm `fs-spkmeans` (frequency sensitive `spkmeans`) that is an extension of `spkmeans` and is applicable to static data that can be read as many times as necessary (Algorithm 4).

Though the algorithm `fs-spkmeans` is motivated by the FSCL, it does not have the incremental flavor of FSCL. To study the effect of the incremental behavior of `fs-spkmeans`, we present and empirically evaluate two variants of this algorithm. The first, called `pifs-spkmeans` (*partially incremental* `fs-spkmeans`), basically incorporates step 2b of `fs-spkmeans` into step 2a. In other words, in each iteration $t$, as soon as a point gets assigned to the $h$-th cluster, the value of $n_h^{(t)}$ is updated. The algorithm is presented in Algorithm 5.

The second variant is called `fifs-spkmeans` (*fully incremental* `fs-spkmeans`), and has the full flavor of competitive learning. In this scheme, in a particular *epoch*,

**Algorithm 4** Frequency Sensitive `SPKMeans` (`fs-spkmeans`)

Set iteration count $t \leftarrow 0$. Choose $k$ points (unit vectors) as the cluster means $\boldsymbol{\mu}_h^{(0)}$, set $n_h^{(0)} \leftarrow \frac{n}{k}$, $h = 1, \cdots, k$.
**repeat**
  {The Assignment Step}
  Set $\mathcal{X}_h \leftarrow \emptyset$
  **for** $i = 1$ to $n$ **do**
    $\mathcal{X}_{h^*} \leftarrow \mathcal{X}_{h^*} \cup \{\mathbf{x}_i\}$, where where

$$h^* = \operatorname*{argmax}_h \frac{1}{n_h^{(t)}} \left\{ \mathbf{x}^T \boldsymbol{\mu}_h + 1 - \frac{n_h}{(n/k)d} \log n_h^{(t)} \right\}$$

  **end for**
  {The Re-estimation Step}
  **for** $h = 1$ to $k$ **do**
    $n_h^{(t+1)} \leftarrow |\{\mathbf{x} : \mathbf{x} \in f_h^{(t)}\}|$
    $\boldsymbol{\mu}_h^{(t+1)} \leftarrow (\sum_{\mathbf{x} \in f_h^{(t)}} \mathbf{x}) / \| \sum_{\mathbf{x} \in f_h^{(t)}} \mathbf{x} \|$
  **end for**
  $t \leftarrow (t+1)$
**until** *convergence*

i.e., a run through all the data points, as soon as a point gets assigned to the $h$-th cluster, both $n_h$ and $\boldsymbol{\mu}_h$ are updated. Thus, in this scheme, we have shrinking as well as moving vMF distributions trying to model the data. The basic algorithm is presented in Algorithm 6.

Note that in both the incremental algorithms, after each point is assigned to a cluster and its count incremented, a constant $1/k$ is subtracted from each $n_h$. This ensures that at any point of time, the total number of points in all the clusters add up to $n$.

## 5.5 Algorithm for Streaming Data

The algorithms presented in section 5.4 necessarily need to know the number of data points to be processed from beforehand. They also need to make multiple read accesses over all the data points. Note that neither of these conditions is sat-

**Algorithm 5** Partially Incremental Frequency Sensitive `SPKMeans` (`pifs-spkmeans`)

Set iteration count $t \leftarrow 0$. Choose $k$ points (unit vectors) as the cluster means $\boldsymbol{\mu}_h^{(0)}$, set $n_h^{(0)} \leftarrow \frac{n}{k}$, $h = 1, \cdots, k$.
**repeat**
  {The Assignment Step}
  Set $\mathcal{X}_h \leftarrow \emptyset$
  **for** $i = 1$ to $n$ **do**
    $\mathcal{X}_{h^*} \leftarrow \mathcal{X}_{h^*} \cup \{\mathbf{x}_i\}$, where

$$h^* = \underset{h}{\operatorname{argmax}} \frac{1}{n_h^{(t)}} \left\{ \mathbf{x}^T \boldsymbol{\mu}_h + 1 - \frac{n_h}{(n/k)d} \log n_h^{(t)} \right\}$$

    $n_{h^*}^{(t)} \leftarrow n_{h^*}^{(t)} + 1$ ; $n_h^{(t)} \leftarrow n_h^{(t)} - \frac{1}{k}$, $\forall h$
  **end for**
  {The Re-estimation Step}
  **for** $h = 1$ to $k$ **do**
    $\boldsymbol{\mu}_h^{(t+1)} \leftarrow (\sum_{x \in f_h^{(t)}} x) / \| \sum_{x \in f_h^{(t)}} x \|$
    $n_h^{(t+1)} \leftarrow n_h^{(t)}, h = 1, \cdots, k, \quad t \leftarrow (t+1)$
  **end for**
**until** *convergence*

isfied when the application demands clustering of streaming data. Streaming data is often typical of non-stationary environments requiring continuous on-line adaptation [RG98]. The need for clustering streaming, normalized data is encountered, for example, for real-time incremental grouping of news stories or message alerts that are received on-line. In classical pattern recognition, streaming data are often encountered in the form of non-linear time series [GW93]. It is not surprising that much work on analyzing streaming data has been done in the neural network, signal processing and applied physics communities, starting from the early days of ADALINE [WL90, Moz93, SDN87, SG97]. More recently, some machine learning researchers have also got interested in this problem [Tur96, HSD01]. But to date, there has been little work on clustering of such data [GMMO00, GG01], and there is a lack of benchmark data sets for the same.

An algorithm working on streaming data gets to read the data only once.

**Algorithm 6** Fully Incremental Frequency Sensitive `SPKMeans` (`fifs-spkmeans`)

Choose $k$ points (unit vectors) as the cluster means $\boldsymbol{\mu}_h$, set $n_h \leftarrow \frac{n}{k}, h = 1, \cdots, k$.
**repeat**
  {The Assignment and Re-estimation Steps}
  Set $\mathcal{X}_h \leftarrow \emptyset$
  **for** $i = 1$ to $n$ **do**
    $\mathcal{X}_{h^*} \leftarrow \mathcal{X}_{h^*} \cup \{\mathbf{x}_i\}$, where

$$h^* = \arg\max_h \frac{1}{n_h} \left\{ \mathbf{x}^T \boldsymbol{\mu}_h + 1 - \frac{n_h}{(n/k)d} \log n_h \right\}$$

    $n_{h^*} \leftarrow n_{h^*} + 1$
    **for** $h = 1$ to $k$ **do**
      $n_h \leftarrow n_h - \frac{1}{k}, \forall h$
      $\boldsymbol{\mu}_h \leftarrow (\boldsymbol{\mu}_h + \frac{1}{n_h}(x_i - \boldsymbol{\mu}_h))/\|\boldsymbol{\mu}_h + \frac{1}{n_h}(x_i - \boldsymbol{\mu}_h)\|$
    **end for**
  **end for**
**until** *convergence*

Thus none of the algorithms presented in section 5.4 can be applied to streaming data. In this section, we present `sfs-spkmeans` (streaming `fs-spkmeans`), a variant of `fs-spkmeans` that can be applied to streaming data since it does not need to know the number of points to be processed and reads every data-point exactly once. Note that the online version may actually be more applicable in certain real life scenarios, e.g., when data is being collected incrementally over time, or, when the clustering has to be done by making a single pass over the data kept in a database.

For constructing the online variant, we first note that a non-normalized mean $\mu^{(t+1)}$ of $(t+1)$ data points can be written as a recursion in terms of $\mu^{(t)}$ [Tra91] as follows:

$$\mu^{(t+1)} = \mu^{(t)} + \frac{1}{t+1}(\mathbf{x}_{t+1} - \mu^{(t)}). \tag{5.14}$$

If the data is obtained from a stationary process, i.e., the parameters of the underlying generative model does not change with time, then $\mu^{(t)}$, as computed by the above recursion will converge, and do not need updating after sufficiently large $t$. However, typical streaming data is non-stationary. There are two popular approaches taken in

such cases: (i) if the data characteristics change abruptly, then such breakpoints can be detected, and a model is fitted for each segment (regime) between two successive breakpoints, assuming stationarity within such segments. Piecewise autoregressive modeling is an example of such an approach. (ii) If the data characteristics vary slowly over time, the problem may be addressed by discounting the past – an approximation recursion can be used that keeps an exponentially decaying window over the history of observations and maintains the effective count $c_{t+1}$ of the history rather than the exact $(t+1)$. More precisely, the approximate recursion for the mean [Tra91] is given by:

$$\tilde{\mu}^{(t+1)} = \tilde{\mu}^{(t)} + \frac{1}{c_{t+1}}(\mathbf{x}_{t+1} - \tilde{\mu}^{(t)}),$$

where $c_{t+1} = (1 - 1/L)c_t + 1$ and $L$ is a large number [Tra91, RG99, NH98]. Note that this exponential decay factor of $(1 - 1/L)$ ensures that $c_{t+1}$ converges from below to $L$. Thus, after the "cold start" period is over, the history maintained in the computation has an effective length $L$. The choice of $L$ depends on the degree of non-stationarity, and a fundamental tradeoff between resolution and memory depth is encountered [PKC94]. We take a similar approach for approximating the normalized mean. The normalized mean of $(t+1)$ data points can be written as a normalized recursion as follows:

$$\boldsymbol{\mu}^{(t+1)} = \frac{\boldsymbol{\mu}^{(t)} + \frac{1}{L_{t+1}}\left\{\mathbf{x}_{t+1} - (L_{t+1} - L_t)\ \boldsymbol{\mu}^{(t)}\right\}}{\|\boldsymbol{\mu}^{(t)} + \frac{1}{L_{t+1}}\left\{\mathbf{x}_{t+1} - (L_{t+1} - L_t)\ \boldsymbol{\mu}^{(t)}\right\}\|}, \tag{5.15}$$

where $L_t = \|\sum_{i=1}^{t} \mathbf{x}_i\|$. Again, using $L_t$ in this recursion results in similar issues as outlined above for the non-normalized case. Using the same exponential decay approximation, we have

$$\tilde{L}_{t+1} = (1 - 1/L)\tilde{L}_t + \|x_{t+1}\| = (1 - 1/L)\tilde{L}_t + 1, \tag{5.16}$$

since $\|x_{t+1}\| = 1$, and $\tilde{L}_0 = 0$. It can be easily seen that $\lim_{t\to\infty} L_t = L$. A direct calculation using the recursion above shows that

$$\tilde{L}_{t+1} - \tilde{L}_t = (1 - 1/L)^t. \tag{5.17}$$

Now, for a large $L$ and $t \ll L$, from (5.17) we have $\tilde{L}_{t+1} - \tilde{L}_t \approx 1$. Using this and the approximations of (5.16), following (5.15), the approximate normalized recursion for $\boldsymbol{\mu}^{(t)}$ is given by

$$\tilde{\boldsymbol{\mu}}^{(t+1)} = \frac{\tilde{\boldsymbol{\mu}}^{(t)} + \frac{1}{\tilde{L}_{t+1}}(\mathbf{x}_{t+1} - \tilde{\boldsymbol{\mu}}^{(t)})}{\|\tilde{\boldsymbol{\mu}}^{(t)} + \frac{1}{\tilde{L}_{t+1}}(\mathbf{x}_{t+1} - \tilde{\boldsymbol{\mu}}^{(t)})\|}, \tag{5.18}$$

where $\tilde{L}_{t+1}$ is given by (5.16).

To make the frequency sensitive version of spherical kmeans applicable to streaming data, as before, we want to make $\kappa_h \propto 1/n_h$. However, the number of points to be processed, $n_h$ is unknown and may be unbounded. Therefore, we use the same exponential decay recursion for $n_h$ so that $\tilde{n}_h^{(t+1)} = (1 - 1/L)\tilde{n}_h^{(t)} + 1$ and $\tilde{n}_h^{(0)} = 0$. Note that the recursion and the base case for $\tilde{n}_h^{(t)}$ and $\tilde{L}_t$ are exactly the same so that, for notation we can use only one of them. We choose to use $\tilde{n}_h^{(t)}$. Then, if $x_{t+1}$ is assigned to cluster $h$, from (5.18), we have

$$\tilde{\boldsymbol{\mu}}_h^{(t+1)} = \frac{\tilde{\boldsymbol{\mu}}_h^{(t)} + \frac{1}{\tilde{n}_h^{(t+1)}}(x_{t+1} - \tilde{\boldsymbol{\mu}}_h^{(t)})}{\|\tilde{\boldsymbol{\mu}}_h^{(t)} + \frac{1}{\tilde{n}_h^{(t+1)}}(x_{t+1} - \tilde{\boldsymbol{\mu}}_h^{(t)})\|}, \tag{5.19}$$

Now, in the limit, all the clusters will have a perfect balancing with effectively $L$ points per cluster and hence $L$ plays the role of $n/k$ in the static case. Thus, following the static case, the proportionality constant in $\tilde{\kappa}_h^{(t)} \propto 1/\tilde{n}_h^{(t)}$ is set to $Ld^2/2$. Since $Ld^2/2n_h \gg d$, following the previous argument and replacing $n/k$ with $L$, the most

likely distribution to have generated a particular point $\mathbf{x}_i$ is given by

$$h^* = \arg\max_h \frac{1}{\tilde{n}_h^{(t)}} \left\{ \mathbf{x}_i^T \tilde{\boldsymbol{\mu}}_h^{(t)} + 1 - \frac{\tilde{n}_h^{(t)}}{Ld} \log \tilde{n}_h^{(t)} \right\}. \tag{5.20}$$

Using the above equation, we present `sfs-spkmeans` in Algorithm 7, the variant of frequency sensitive spherical kmeans applicable to streaming data .

**Algorithm 7** Streaming Frequency Sensitive `SPKMeans` (`sfs-spkmeans`)

Set data count $t \leftarrow 0$. Choose $k$ points (unit vectors) as the cluster means $\tilde{\boldsymbol{\mu}}_h^{(0)}$, set $\tilde{n}_h^{(0)} \leftarrow 0$, $h = 1, \cdots, k$.
**for** the next data-point $\mathbf{x}_{t+1}$ **do**
  Assign $\mathbf{x}_{t+1}$ to the cluster $\mathcal{X}_{h^*}^{(t)}$ where

$$h^* = \arg\max_h \frac{1}{\tilde{n}_h^{(t)}} \left\{ \mathbf{x}_{t+1}^T \tilde{\boldsymbol{\mu}}_h^{(t)} + 1 - \frac{\tilde{n}_h^{(t)}}{Ld} \log \tilde{n}_h^{(t)} \right\}$$

  $\tilde{n}_{h^*}^{(t+1)} \leftarrow (1 - 1/L)\tilde{n}_{h^*}^{(t)} + 1$
  $\tilde{\boldsymbol{\mu}}_{h^*}^{(t+1)} \leftarrow \frac{(\tilde{\boldsymbol{\mu}}_{h^*}^{(t)} + \frac{1}{\tilde{n}_{h^*}^{(t+1)}}(\mathbf{x}_{t+1} - \tilde{\boldsymbol{\mu}}_{h^*}^{(t)}))}{\|\tilde{\boldsymbol{\mu}}_{h^*}^{(t)} + \frac{1}{\tilde{n}_{h^*}^{(t+1)}}(\mathbf{x}_{t+1} - \tilde{\boldsymbol{\mu}}_{h^*}^{(t)})\|}$
  **for** $h = 1$ to $k$, $h \neq h^*$ **do**
    $\tilde{n}_h^{(t+1)} \leftarrow \tilde{n}_h^{(t)}$
    $\tilde{\boldsymbol{\mu}}_h^{(t+1)} \leftarrow \tilde{\boldsymbol{\mu}}_h^{(t)}$
  **end for**
  $t \leftarrow (t + 1)$
**end for**

**Computational Requirements:** Finally, a word on the complexity of the proposed approaches. Since all of the proposed approaches follow the basic infrastructure of `kmeans`, each iteration is linear in the number of data-points and the number of clusters. For the static algorithms, under realistic assumptions [BDGS03], the algorithms converge within a finite number of iterations. Note that, if need be, the approximations can be totally avoided with some extra computational effort [BDGS03]. For the streaming data, since the data points are processed one at a time, the algorithm is of course linear in the number of data points, but in a rather

different sense. Further, for each data point, the streaming algorithm is linear in the number of clusters. Hence, all the proposed approaches are very scalable and can be employed in large scale clustering tasks.

## 5.6 Experimental Results

In this section, experimental results on the proposed ideas are described. We present results on three high-dimensional text datasets - the Classic3 dataset[3], the News 20 dataset[4] and the Yahoo news dataset[5](K1) for the empirical performance analysis. In spite of being high-dimensional, sparse datasets, they have quite different properties that are quite useful in demonstrating the biases of the proposed algorithms.

**The Classic3 dataset** contains 3893 files, among which 1400 Cranfield documents are from aeronautical system papers, 1033 Medline documents are from medical journals, and 1460 Cisi documents are from information retrieval papers. The toolkit MC [DFG01] was used for creating the high-dimensional vector space model for the text documents and a total of 6061 words were used. Thus, each document, after normalization, is represented as a unit vector in a 6061 dimensional space. This is a relatively simple dataset in the sense that the documents in the 3 clusters are on completely different topics. Also, the *natural clusters* [6] are quite balanced.

**The News20 dataset** is a collection of 19,997 messages, collected from 20 different usenet newsgroups, with approximately 1000 messages per newsgroup. chosen at random and partitioned by newsgroup name. The headers for each of the mes-

[3]http://www.cs.utexas.edu/users/yguan/datamining/project/project.html

[4]http://www.ai.mit.edu/people/jrennie/20Newsgroups/

[5]ftp://ftp.cs.umn.edu/users/boley/PDDPdata/

[6]All three data sets used have class labels. We call the clustering indicated by the class labels as the "natural clustering" of the dataset. Evaluating clustering quality by comparing cluster labels with class labels is quite common, but such results should be presented with a caveat, since some classes can be multi-modal, and classes may overlap as well, as is quite evident in the News20 dataset.

sages were removed so that they do not bias the results. Using the toolkit MC, the high-dimensional model had a total of 26099 words. This is a typical dataset that one may encounter in real life — it is very high-dimensional, sparse and there is significant overlaps between the newsgroups. In fact, some cross-posted articles appear multiple times in the dataset – once under every group in which it was posted. Another feature of this dataset is that the natural clusters are perfectly balanced.

**The Yahoo20 dataset** (K-series) is a collection of 2340 Yahoo news articles belonging one of 20 different Yahoo categories. The K1 set actually gives the high-dimensional vector space model having 21839 words. After normalization, the data points reside on the surface of a 21839 dimensional hypersphere. The salient feature of this dataset is that the natural clusters are not at all balanced, with cluster sizes ranging from 9 to 494. This is where it significantly differs from the two previous datasets. This dataset helps in studying the effect of the balancing bias of the proposed algorithms on the cluster quality if the natural clusters are highly unbalanced.

### 5.6.1 Performance Measures

We study four performance measures to get a comparative understanding of the proposed algorithms – two of them measure cluster quality whereas the other two measure cluster balancing.

**Cluster Quality** is evaluated by an external and an internal measure. External measures such as purity and entropy of clusters [SGM00] can be used if external information such as class labels for all data points can be obtained [Gho03]. An increasingly popular external measure is the mutual information between the cluster assignments and a pre-existing labeling of the dataset. Formally, if $X$ is a random variable for the cluster assignments and $Y$ is a random variable for the pre-existing labels on the same data, then the mutual information $I(X;Y) = E_{X,Y}[\ln \frac{p(X,Y)}{p(X)p(Y)}]$

is the amount of statistical information shared by $X$ and $Y$ [CT91]. If $m_{hl}$ is the number of documents in the $h$-th cluster, $h \in \{1, \cdots, k\}$, that has class label $l \in \{1, \cdots, c\}$, then the empirical estimate of the probability of the joint event $\{X = h, Y = l\}$ is computed as $p(X = h, Y = l) = m_{hl}/n$, where $n$ is the total number of documents. The mutual information is computed using the empirical estimates for the joint events and the corresponding empirical marginals. We shall use a *normalized mutual information* (NMI) measure so that the numbers are in the range $[0, 1]$. The normalization is done using the arithmetic mean of the maximum possible entropies of the empirical marginals, i.e., $\text{NMI}(X, Y) = \frac{I(X,Y)}{(\log k + \log c)/2}$. The NMI [SG02] measures the amount of statistical similarity between the cluster assignments and the pre-existing labels under an appropriate (constant) scaling. This measure is better than certain other commonly used external measures such as entropy or purity [SGM00], in the sense that NMI does not necessarily increase with increase in the number of clusters $k$, whereas both entropy and purity do. We shall investigate another objective way of evaluating clustering in Chapter 7.

As a second point of reference, an internal measure of cluster quality, as indicated by the `spkmeans` *objective function* (SOF) value (5.10), is used. Note that using this measure favors `spkmeans` which optimizes this measure, while all the proposed methods attempt to optimize modified versions of this objective that also weave in balancing constraints.

**Cluster Balancing** is also evaluated by two measures. One measure is the *standard deviation in cluster sizes* (SDCS) for a given number of clusters requested from the algorithm. As mentioned earlier, obtaining balanced clusters, i.e., clusters with approximately equal sized clusters, is often an application requirement, or a desirable regularization property. SDCS is one measure that helps in understanding the balancing behavior of a clustering algorithm. Thus, if $\{n_1, \cdots, n_k\}$ are the sizes of the $k$ clusters generated by an algorithm, then $\text{SDCS} = \{\frac{1}{k-1}\sum_{h=1}^{k}(n_h - \frac{n}{k})^2\}^{1/2}$.

In addition, it is also useful to know whether an algorithm is returning empty or extremely small clusters. To quantify this behavior, the second measure we use is the ratio of the minimum cluster size generated by an algorithm to the expected cluster size under perfect balancing. We shall refer to this measure as *ratio of minimum to expected* (RME). By definition, RME = $(\min\{n_1, \cdots, n_k\})/(n/k)$.

### 5.6.2 Experiments with Static Algorithms

In this section, we present results on the performance of the proposed static algorithms — `fs-spkmeans`, `pifs-spkmeans` and `fifs-spkmeans` — and compare them with that of the basic `spkmeans` algorithm[7]. We study the algorithms over a reasonable (depending on the dataset under consideration) range of values for the number of clusters to get a better understanding of their properties. All the results presented are averaged over 10 runs. The initial $k$ means of the spherical kmeans were generated by computing the mean of the entire data and making $k$ small random perturbations to this mean [DFG01]. For stability and repeatability, the frequency sensitive algorithms were initialized at points of local minima of the `spkmeans` objective function.

#### The Classic3 dataset

On the Classic3 dataset, all the algorithms perform quite similarly in terms of the two cluster quality measures. All the four algorithms achieve their individual highest values of NMI at $k = 3$, which is the number of natural clusters in the data (Figure 5.1(a)). Even for other values of $k$, they perform quite similarly in terms of NMI, though `pifs-spkmeans` seems to be perform a little differently from the rest of the group. The SOF values for all the algorithms are almost the same (Figure 5.1(b)).

[7] We do not compare with `kmeans` or equivalent algorithms since they perform miserably on high dimensional text data [SGM00]

A difference in their performance is observed while studying the cluster balancing measures. The `pifs-spkmeans` algorithm gives a much lower SDCS as compared to the other algorithms (Figure 5.1(c)). Also, it gives a much higher RME as compared to the other algorithms (Figure 5.1(d)). This points out to the fact that `pifs-spkmeans` has high bias towards balancing.

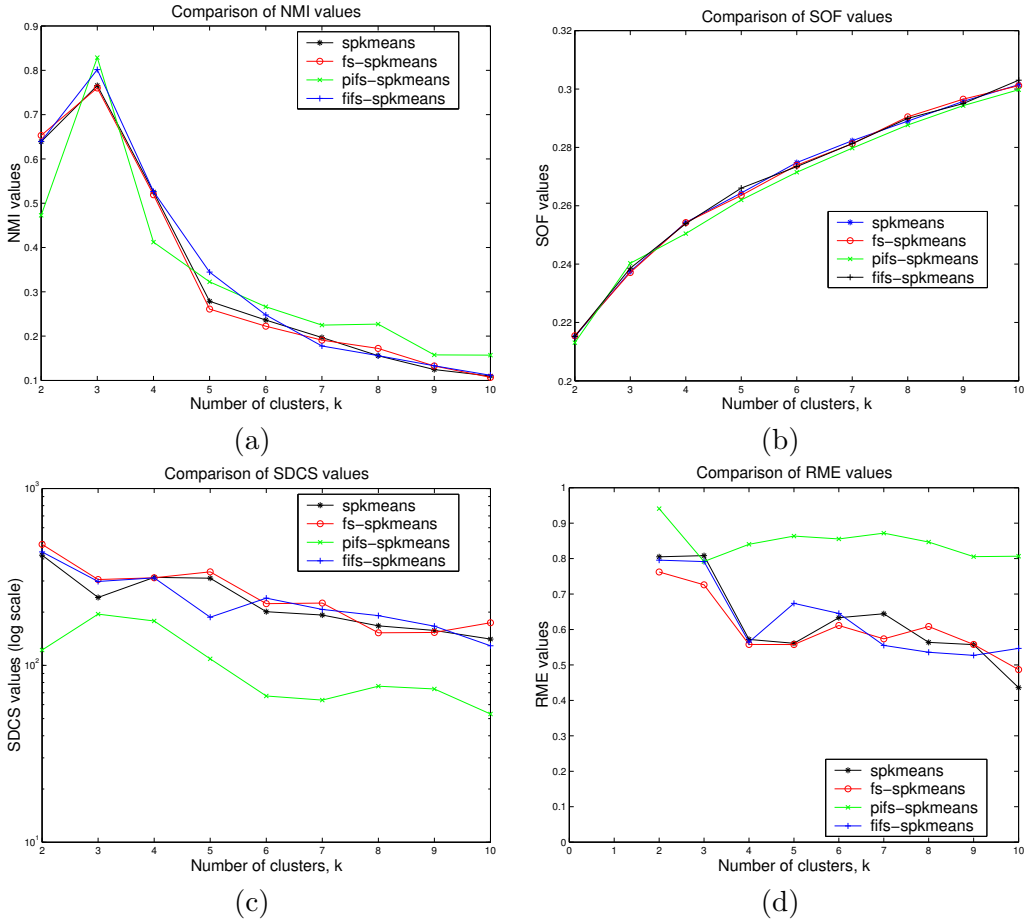


Figure 5.1: Comparison between the static algorithms on the Classic3 data: (a) the normalized mutual information values, (b) the `spkmeans` objective function values, (c) the standard deviation in cluster sizes, and (e) the ratio of the minimum to expected cluster size values, averaged over 10 runs of each algorithm.

### The News20 dataset

The News20 dataset demonstrates the basic nature of the four algorithms quite well. In all the experiments, and in terms of all the performance measures, `fs-spkmeans` and `fifs-spkmeans` show very similar behavior. As seen before in the Classic3 dataset, `pifs-spkmeans` has a high bias towards balancing, whereas `spkmeans` has no explicit mechanism for balancing.

All the algorithms achieve their individual highest values of the NMI at $k = 20$, which is the correct number of clusters (Figure 5.2(a)). At $k = 20$, `fifs-spkmeans` and `fs-spkmeans` perform better than the other two in terms of the NMI, and also show good balancing. For lower values of $k$, `pifs-spkmeans` performs worse than the other three which have quite similar behavior. For much higher values of $k$, `spkmeans` has significantly higher values of NMI compared to the three proposed approaches, since it starts generating zero-sized clusters (Figure 5.2(d)) in order to maintain the NMI and objective at a reasonable value. On the other hand, since none of the proposed algorithms generate zero sized clusters, their performance in terms of NMI suffers. As seen from Figures 5.2(c),(d), `pifs-spkmeans` has the most bias towards balancing thereby achieving the lowest SDCS and the highest RME values for the entire range of $k$ over which experiments were performed. It is interesting to note that `fs-spkmeans` and `fifs-spkmeans` seems to follow a middle ground in terms of its cluster balancing and quality biases. It is also to be noted that the SOF values for the proposed algorithms are equal or greater than those achieved by `spkmeans`.

### The Yahoo20 dataset

As mentioned earlier, the Yahoo20 dataset is highly unbalanced according to the labelling it has. Hence, results on this dataset show how the proposed algorithms handle the data when their balancing bias is not going to help.

It is interesting to see that the performance of the algorithms in terms of the NMI is quite similar to what was observed for the News20 dataset. As before `fs-spkmeans` and `fifs-spkmeans` perform very similarly and the NMI values they achieve deteriorate for values of $k$ greater than 20, the correct number of clusters (Figure 5.3(a)). `pifs-spkmeans` performs poorly in terms on the NMI because of its high bias towards balancing that does not help in this particular dataset. It also performs slightly worse than the other algorithms in terms of the SOF values (Figure 5.3(b)). However, as before, it consistently gives the lowest SDCS (Figure 5.3(c)) and highest RME values (Figure 5.3(d)). `spkmeans` maintains a reasonable value of the NMI even for large values of $k$ by generating empty clusters. It is interesting to note that due to fact that the natural clusters are not at all balanced, `fs-spkmeans` and `fifs-spkmeans` give quite low values of RME, but never actually give a zero-sized cluster in the range of $k$ over which experiments were performed. Again, these two algorithms seem to have a good balance between the biases and can respond quite well to the underlying nature of the dataset.

### Summary of Results

Both `fs-spkmeans` and `fifs-spkmeans` perform admirably when the value of $k$ chosen is in the neighborhood of the number of classes in the data. They are comparable to or superior than `spkmeans` in terms of cluster quality, and superior in terms of balancing. This result is particularly remarkable for the Yahoo20 dataset where the underlying classes have widely varying priors. This is indicative of the beneficial effect of the regularization provided by the soft balancing constraint. However, if $k$ is chosen to be much larger than the number of natural clusters, `spkmeans` has an advantage since it starts generating zero-sized clusters, while the others are now hampered by their proclivity to balance cluster sizes. On the other hand, if balancing is very critical, then `pifs-spkmeans` is the best choice, but it has to compromise

to some extent on cluster quality in order to achieve its superior balancing. So the choice of algorithm clearly depends on the nature of the dataset and the clustering goals, but in general, both `fs-spkmeans` and `fifs-spkmeans` are attractive even when balancing is not an objective.

### 5.6.3 Experiments with the Streaming Algorithm

The primary problem in experimenting with the streaming algorithm is that there is no well-known benchmark for clustering of high-dimensional, normalized streaming data. So, the experiments with the streaming algorithms were done by artificially "streaming" the public domain static datasets. The data points are presented sequentially to the `sfs-spkmeans` algorithm, repeating the process as many times as necessary in order to simulate streaming data. We call the sequence of showing every document in the selected dataset once as an *epoch* and the algorithm is run over multiple epochs until it converges or some preset maxEpoch value is reached. Note that the resulting scheme is very similar to `fifs-spkmeans` but there are subtle differences. In order to understand the effect of the choice of $L$, the number to which the norm and the effective cluster sizes of `sfs-spkmeans` converges, we present results corresponding to two choices of $L$: 100 and 1000, and the corresponding algorithms will be referred to as `sfs100-spkmeans` and `sfs1000-spkmeans` respectively. Note that both these values of $L$ are less than the data set sizes. This means that `sfs-spkmeans` has less effective memory than the static algorithms. In fact, such a low effective memory handicaps the streaming algorithm as compared to the static ones which use all the data to update their parameters. As we shall see, the streaming algorithm actually performs reasonably well even with this handicap.

There are three aspects to be considered when evaluating a streaming algorithm [WS85]: (i) how quickly does it ramp up to a solution, (ii) the quality of the solution, and (iii) if the data characteristics change, how quickly does the system

respond to such changes. Since in this chapter, the streaming data sets are obtained by reproducing a fixed data set end-to-end, all the solutions show an asymptotic behavior. So we shall first address aspect (ii) and compare these asymptotic solutions with those obtained by the static algorithms. Then, in the next subsection, we address aspect (i) and look at the learning curves.

### Asymptotic Results

The streaming algorithm ramps up to a "steady state" solution in a few epochs and after that there are only minor perturbations to this solution. In this section, such steady state solutions are averaged over 10 runs and then compared with spkmeans and fs-spkmeans.

**The Classic3 dataset:** For the Classic3 dataset, all the algorithms achieve their highest individual values of NMI at $k = 3$, the actual number of clusters in the dataset. The streaming algorithms achieve higher NMI at $k = 3$ compared to the batch algorithms (Figure 5.4(a)). All the four algorithms perform very similarly in terms of the NMI for all other values of $k$. The SOF values achieved by the static algorithms are consistently higher than that by the streaming algorithms (Figure 5.4(b)). There is no significant difference between the behavior of the algorithms in terms of the two cluster balancing measures (Figures 5.4(c),(d)).

**The News20 dataset:** In the 20 News dataset, the streaming algorithms perform significantly better than the static ones in terms of the NMI (Figure 5.5(a)). The primary reason for this is that since the natural clusters in the data are perfectly balanced and the streaming algorithms are biased towards balanced clustering, they get the correct structure in the data due to their bias. Among the streaming algorithms, `infs100-spkmeans` performs marginally better than `infs1000-spkmeans` though the differences are not always significant. The SOF values for the static algorithms are significantly better than those achieved by the streaming algorithms

(Figure 5.5(b)). There is no significant difference in the SDCS for the various algorithms (Figure 5.5(c)). The frequency sensitive algorithms perform better than `spkmeans` in terms of the RME values, and the streaming algorithms give higher values of RME than `fs-spkmeans`(Figure 5.5(d)).

**The Yahoo20 dataset:** In the Yahoo dataset, the static algorithms seem to achieve higher values of NMI than the streaming ones (Figure 5.6(a)). In the trade-off between balancing and cluster quality, the streaming algorithms seem to give more importance to the balancing aspect whereas the static ones seems to give higher priority to the cluster quality. The streaming algorithms being biased towards the balancing criterion, performs poorly in terms of the NMI in this dataset that has highly unbalanced natural clusters. Due to this bias, they give significantly better RME values as compared to the static algorithms (Figure 5.6(d)). Like the other two datasets, the SOF values achieved by the static algorithms are significantly better than those by the streaming ones (Figure 5.6(b)). Also, just like the other datasets, there is not much difference in the SDCS across all the algorithms (Figure 5.6(c)).

### Learning Curves

To get a better understanding of how quickly `sfs-spkmeans` reaches a steady state solution, we study it closely on three *randomly chosen* runs on the three datasets for different number of clusters. Note that we do not average over multiple runs since then the corresponding epochs of the same run will be lost. The results are presented for both `sfs100-spkmeans`. and `sfs1000-spkmeans` in Figures 5.7 and 5.8 respectively.

Consider Figure 5.7 first. In the Classic3 dataset, the NMI for $k = 3$, the correct number of clusters, shoots up in the 2nd epoch itself and stays at that level from that point onwards till convergence (Figure 5.7(a)). This shows that the data set has a very simple structure and a single epoch was sufficient to capture it. For

$k = 2$ and $k = 4$, the algorithm does not get any structure in the data as shown by the NMI plots. In the News20 dataset, there is an initial increase in NMI for $k = 10, 20, 30$ (Figure 5.7(b)). However, the plot for $k = 30$ plateaus at a much lower value of NMI than that for $k = 10, 20$. It is interesting to note that although the NMI values for $k = 10, 20$ are quite similar in the first few epochs, the algorithm for $k = 20$ eventually finds a better structure in the data — this fact is reflected in the plots as the NMI values for $k = 20$ crosses that for $k = 10$ before convergence. The behavior in the Yahoo dataset is quite similar to that observed for the News20 dataset in terms of the NMI.

In the Classic3 dataset, the SDCS values for $k = 3, 4$ are significantly better than that for $k = 2$ (Figure 5.7(b)). A similar pattern is observed for the other two datasets. Note that we are showing results for three values of $k$ for all the datasets – one value $\underline{k}$ is less than the number of natural clusters in the data, one value $k^*$ is (approximately) equal to that, and the third value $\overline{k}$ is greater than that. The general pattern we observe is that the algorithm performs well in terms of the NMI for $k = \underline{k}, k^*$, and performs well in terms of the SDCS for $k = k^*, \overline{k}$. Thus, the values at or around $k^*$ are always in the group that performs well for both the measures, whereas the performance for the other values of $k$ suffer in one measure or the other. Also, the RME values for $k = k^*$ are higher or at least as good as that for $k = \underline{k}, \overline{k}$. Thus, we note that the algorithm performs the best for the performance measures under consideration at values close to the natural number of clusters, or, in other words, gets the structure in the data quite well. This is a very desirable quality for a clustering algorithm.

A similar behavior is observed for all the datasets while studying the learning curves generated by `infs1000-spkmeans` in Figure 5.8. The number of epochs required to converge is normally higher than that for the $L = 100$ case since the effective learning rate, being inversely proportional to $L$, is much lower in this case.

The NMI values for $k^*$ after convergence is higher than the others for the Classic3 and News20 datasets (Figures 5.8(a),(c)). For the Yahoo dataset, because of its complicated structure, the preset maxEpoch value of 100 is reached before the algorithms converges – the NMI values for $\underline{k}$ and $k^*$ are still increasing and the value for $\underline{k}$ is slightly higher when the epochs are terminated (Figure 5.8(e)). As before, the SDCS values for $k^*, \overline{k}$ are better than that for $\underline{k}$ across all datasets. Also, the RME values for $k^*, \underline{k}$ are better than that for $\overline{k}$. Thus, we once again see that the algorithm tends to have the best balance across all the performance metrics for values of $k$ close to the natural number of clusters.

## 5.7 Discussion

Obtaining a balanced solution is an explicit goal in certain clustering applications irrespective of the underlying structure of the data. More commonly, obtaining clusters of comparable sizes is not a stated objective, but some amount of balancing helps in countering poor initializations in iterative clustering algorithms that converge to only a local optimum. The problem of poor initialization is exacerbated when both the input dimensionality as well as the number of clusters sought are high, thereby vastly expanding the solution space. In addition to incorporating a conscience mechanism to competitive learning, a variety of other approaches have been proposed over the years [MH98] to overcome poor initializations in iterative clustering algorithms that are otherwise attractive because of simplicity, low computational complexity, etc.

In this chapter, we focused on applications where the data is normalized to lie on the surface of a hypersphere. For such datasets, the clustering problem was posed as a maximum likelihood estimation of $k$ vMF distributions that are assumed to have generated the observed data. This generative model can be adapted, if need be, to provide various degrees of balancing. In the process, we derived certain

existing, heuristically proposed algorithms (spkmeans, FSCL) from first principles. The empirical results were also encouraging in that they were are both superior (using external as well as internal criteria) as well as significantly better balanced. This was true even for the highly imbalanced Yahoo20 dataset.

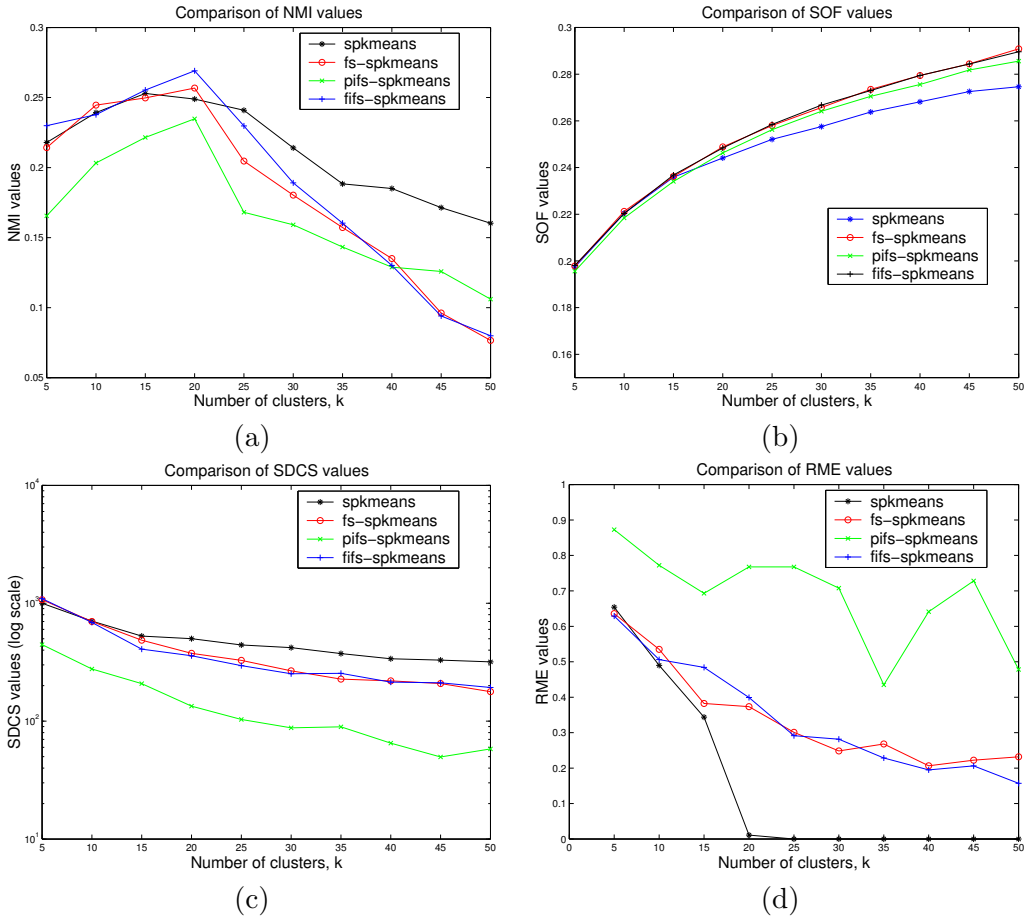


Figure 5.2: Comparison between the static algorithms on the News20 data: (a) the normalized mutual information values, (b) the `spkmeans` objective function values, (c) the standard deviation in cluster sizes, and (e) the ratio of the minimum to expected cluster size values, averaged over 10 runs of each algorithm.

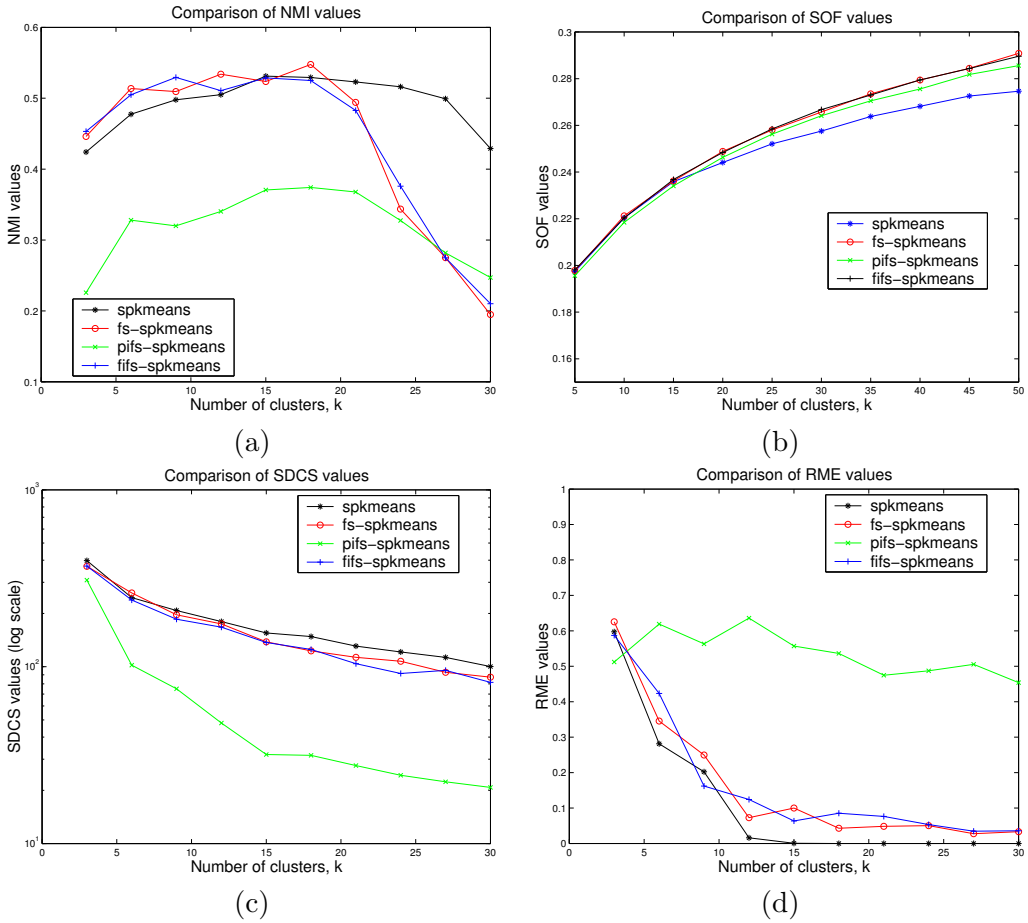


Figure 5.3: Comparison between the static algorithms on the Yahoo20 data: (a) the normalized mutual information values, (b) the `spkmeans` objective function values, (c) standard deviation in cluster sizes, and (e) the ratio of the minimum to expected cluster size values, averaged over 10 runs of each algorithm.

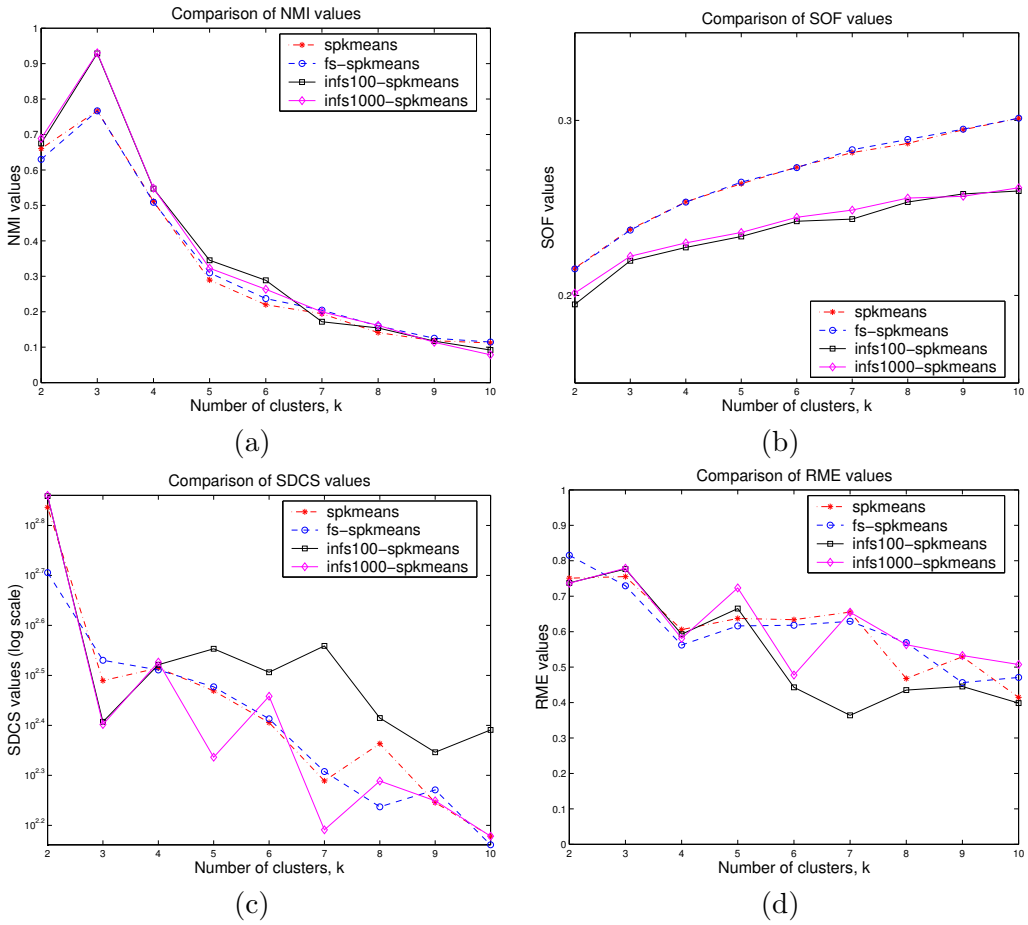


Figure 5.4: Comparison between streaming and static algorithms on the Classic3 data: (a) the normalized mutual information values, (b) the `spkmeans` objective function values, (c) the standard deviation in cluster sizes, and (d) the ratio of minimum to the expected cluster size values, averaged over 10 runs of each algorithm.

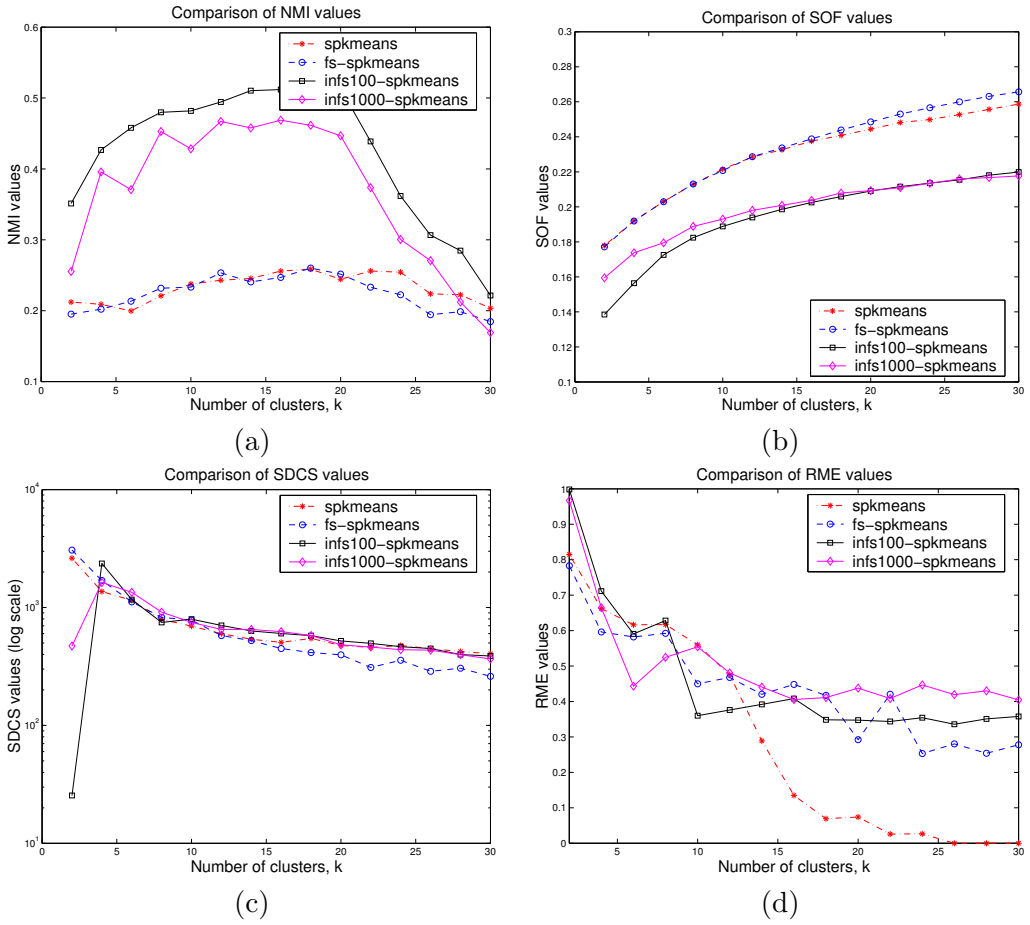


Figure 5.5: Comparison between streaming and static algorithms on the News20 data: (a) the normalized mutual information values, (b) the `spkmeans` objective function values, (c) the standard deviation in cluster sizes, and (d) the ratio of the minimum to the expected cluster size values, averaged over 10 runs of each algorithm.

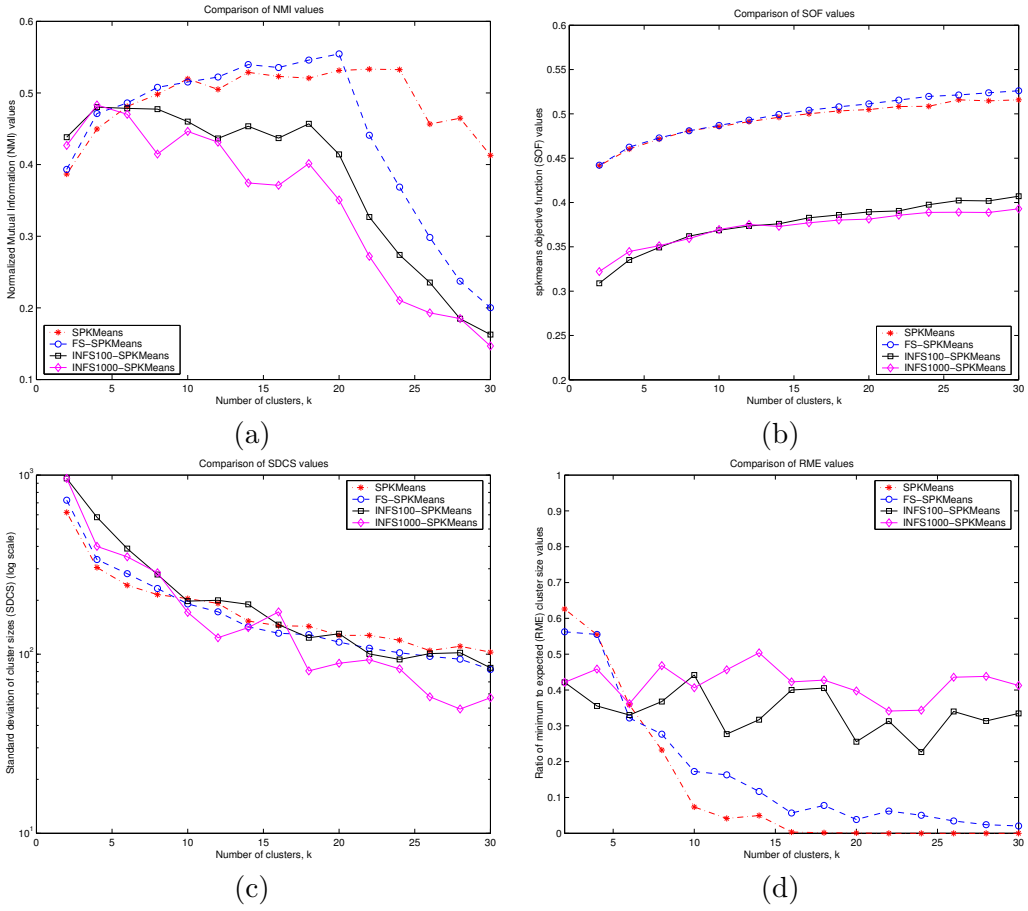


Figure 5.6: Comparison between streaming and static algorithms on the Yahoo20 data: (a) the normalized mutual information values, (b) the `spkmeans` objective function values, (c) the standard deviation in cluster sizes, and (d) the ratio of minimum to the expected cluster size values, averaged over 10 runs of each algorithm.

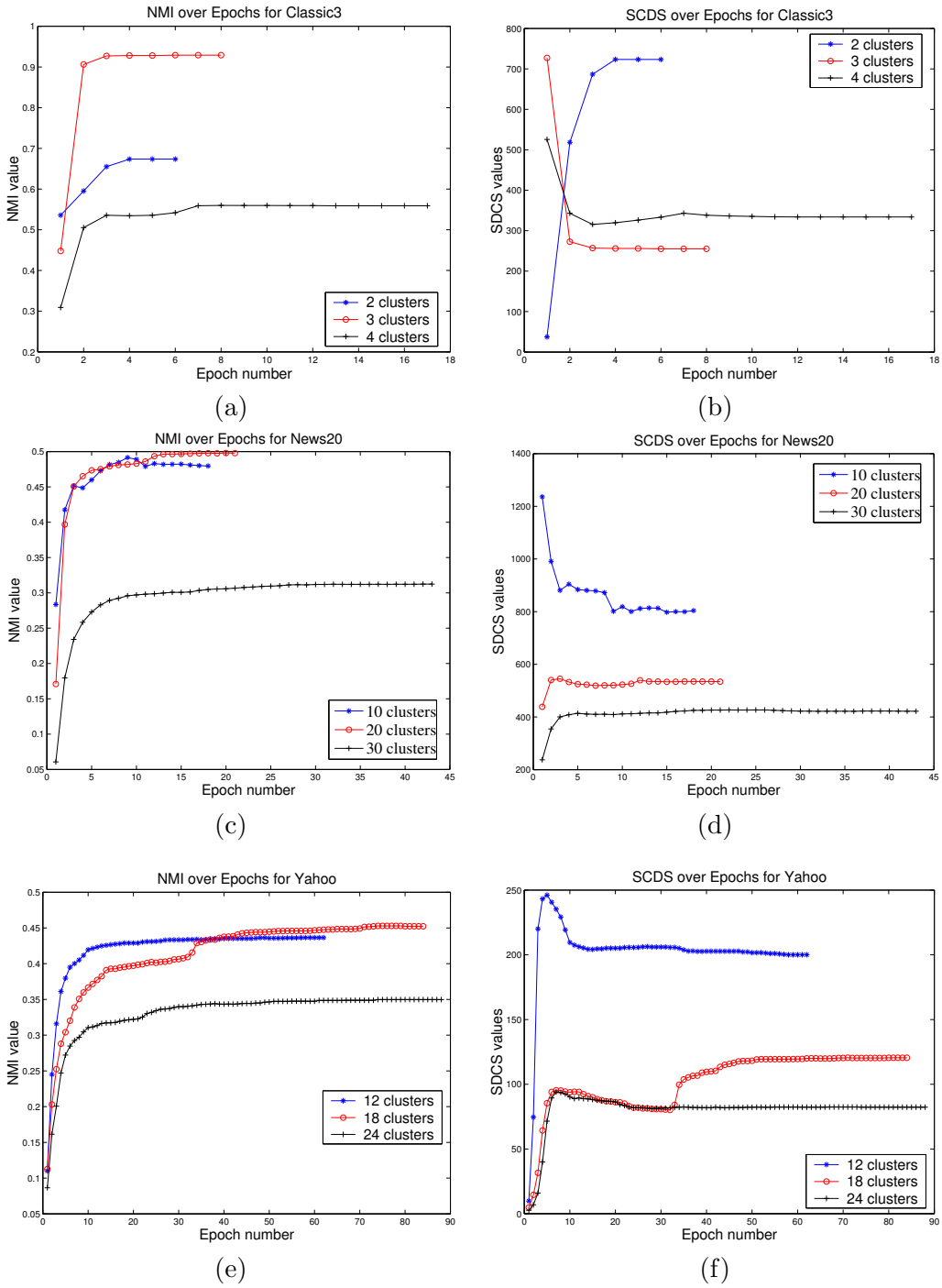


Figure 5.7: Comparison of NMI and SDCS values over epochs (`infs100-spkmeans`) on particular runs for Classic3, News20 and Yahoo20 datasets.

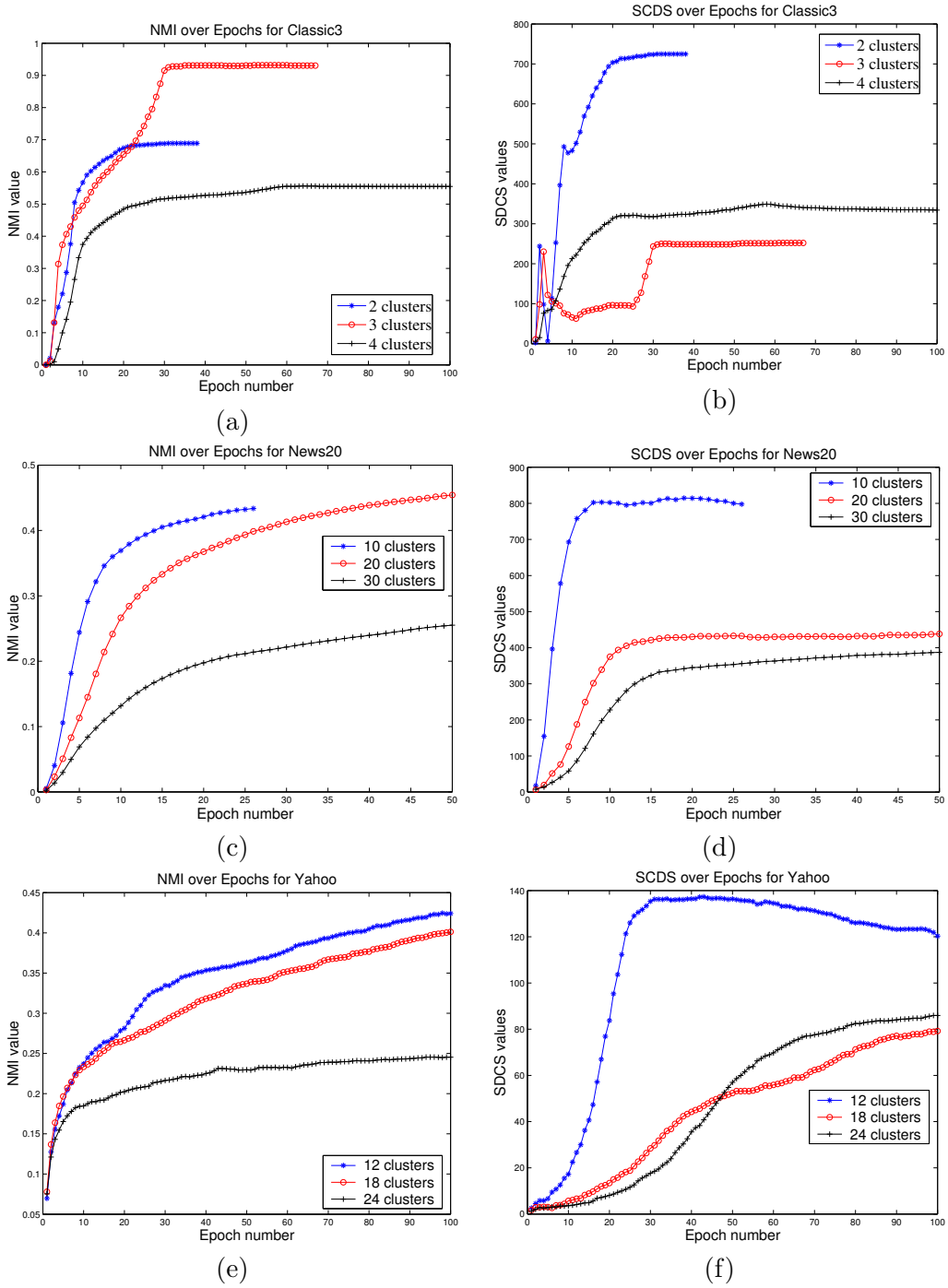


Figure 5.8: Comparison of NMI and SDCS values over epochs (`infs1000-spkmeans`) on particular runs for Classic3, News20 and Yahoo20 datasets.

# Chapter 6

# Scalable Clustering with Balancing Constraints

In this chapter, we address the issue of developing scalable clustering algorithms that satisfy balancing constraints on the cluster sizes. Unlike Chapter 5, where balancing was done without any guarantees on the minimum number of points per cluster, we now focus on algorithms with provable balancing guarantees. An $O(kN \log N)$ algorithm is presented for clustering $N$ data-points into $k$ clusters so that each cluster has at least $m$ points for some given $m \leq \frac{N}{k}$. The proposed scheme can be broken down into three steps - sampling, clustering of samples and populating the clusters while keeping the balance followed by refinements. We address each of the steps separately and show that the three-step process gives a very general methodology for scaling up clustering algorithms while satisfying balancing constraints. A brief overview of the three steps is presented in Section 6.1. The three steps are discussed and analyzed in detail in Sections 6.2, 6.3 and 6.4. A brief discussion of some important issues regarding the proposed scheme is presented in Section 9.3. In Section 6.5 we present experimental results on high-dimensional text clustering problems.

[0]The work presented in this chapter has earlier appeared in part as [BG02b].

## 6.1 Overview

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$, $\mathbf{x}_i \in \mathbb{R}^d, \forall i$, be a set of data-points that needs to be clustered. Let $d : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}_+$ be a given distance function between any two points in $\mathbb{R}^d$. The *clustering problem* we consider is that of finding a disjoint $k$-partitioning $\{S_h\}_{h=1}^k$ of $\mathcal{X}$ and a corresponding set of $k$ cluster representatives $\mathcal{M} = \{\mu_h\}_{h=1}^k$ in $\mathbb{R}^d$ for a given $k$ such that the clustering objective function

$$\mathbb{L}(\{\mu_h, S_h\}_{h=1}^k) = \sum_{h=1}^{k} \sum_{\mathbf{x} \in S_h} d(\mathbf{x}, \mu_h) \tag{6.1}$$

is minimized under the constraint that $|S_h| \geq m, \forall h$, for a given $m$ with $mk \leq N$. Further, we assume that for the *optimal* partitioning $\{S_h^*\}_{h=1}^k$, we have

$$\min_h \frac{|S_h^*|}{N} \geq \frac{1}{l} \tag{6.2}$$

for some integer $l \geq k$. In other words, if samples are drawn uniformly at random from the set $\mathcal{X}$, the probability of picking a sample from any particular optimal partition is at least $\frac{1}{l}$ for some given $l \geq k$. Note that $l = k$ if and only if $|S_h^*| = N/k, \forall h$, so that the optimal partitions are exactly of equal size.

For the clustering problem to be tractable, the distance function $d$ has to be well-behaved in the following sense: Given a set of points $\mathbf{x}_i, \cdots, \mathbf{x}_n$, there should be an efficient way of finding the representative

$$\boldsymbol{\mu} = \operatorname*{argmin}_c \sum_{i=1}^{n} d(\mathbf{x}_i, c) \ .$$

While a large class of distance functions are well-behaved in the above sense, in this article, we focus on the squared Euclidean distance and the cosine distance, for both of which the representative can be computed in $O(n)$ time. As we discussed in Chapter 3, the representative can be computed in $O(n)$ for all Bregman

divergences [BMDG04]. For the general framework discussed in this article, any meaningful distance function can potentially be used as long as the computation of the representative can be done efficiently.

To make the solution of this constrained clustering problem scalable, we break up the solution into three steps making use of the information regarding the nature of the optimal clustering. The three steps in the proposed scheme are as follows:

Step 1 *Sampling of the given data:* The given data is first sampled in order to get a small representative subset of the data. One potential problem with sampling is that one may end up sampling all the points from only a few of the optimal partitions so that the sampled data is not a representative subset for clustering. We show that this kind of a scenario is unlikely under the assumptions of (6.2) and compute the number of samples we must must draw from the original data in order to get a good representation from each of the optimal partitions in the sampled set with high probability.

Step 2 *Clustering of the sampled data:* Any clustering algorithm that fits the problem domain can be used in this stage. The algorithm is run on the sampled set. Since the size of the sampled set is much less than that of the original data, one can use slightly involved algorithms as well, without much blow-up in complexity. It is important to note that the general framework works for a wide class of distance (or similarity) based clustering algorithms as long as there is a concept of a representative of a cluster and the clustering objective is to minimize the average distance (or maximize the similarity) to the corresponding representatives.

Step 3 *Populating and Refining the clusters:* The third step has two parts: Populate and Refine. In the first part, the small clusters generated in the second step are populated with the data-points that were not sampled in the first

step. For populating, we propose an algorithm `poly-stable`, motivated by the stable marriage problem [GI89], that satisfies the requirement on the number of points per cluster, thereby generating a *feasible* and reasonably good clustering.

In the second part, an iterative refinement algorithm `refine` is applied on this stable feasible solution to monotonically decrease the clustering objective function while satisfying the constraints all along. On convergence, the final partitioning of the data is obtained.

Note that populating and refining of the clusters can be applied irrespective of what clustering algorithm was used in the second step, as long as there is a way to represent the clusters. In fact, the third step is the most critical step and the first two steps can be considered a good way to initialize the populate-refine step.

The three steps outlined above provide a very general framework for scaling up clustering algorithms under mild conditions, irrespective of the domain from which the data has been drawn and the clustering algorithm that is used as long as they satisfy the assumptions.

## 6.2 Sampling

First, we examine the question: given the set $\mathcal{X}$ having $k$ underlying optimal partitions $\{S_h^*\}_{h=1}^k$ such that the probability of sampling a point uniformly at random from any particular partition is at least $\frac{1}{l}$, what is the number $n$ of samples that need to be drawn from $\mathcal{X}$ so that there are at least $s \gg 1$ points from each of the $k$ partitions with high probability? Let $X$ be a random variable for the number of samples that need to be drawn from $\mathcal{X}$ to get at least $s$ points from each cluster. $E[X]$ can be computed by an analysis using the theory of recurrent events and renewals [Fel67].

A simpler version of this analysis can be found in the so-called Coupon Collector's problem [MR95], or, equivalently in the computation of the cover time of a random walk on a complete graph. With a similar analysis in this case, we get the following result.

**Lemma 5** *If $X$ is the random variable for the number of samples be drawn from $\mathcal{X}$ to get at least $s$ points from each partition, $E[X] \leq sl \ln k + O(sl)$.*

*Proof:* Let $C_1, C_2, \cdots, C_X$ be the sequence of samples drawn where $C_h \in \{1, \cdots, k\}$ denotes the partition number from which the $i$th sample was drawn. We call the $h$-th sample $C_h$ *good* if less than $s$ samples have been drawn from partition $C_h$ in the previous $(h-1)$ draws. Note that the first $s$ samples are always good. The sequence of draws is divided into epochs where epoch $i$ begins with the draw following the draw of the $i$th good sample and ends with the draw on which the $(i+1)$-th good sample is drawn. Since a total of $sk$ good samples have to be drawn, the number of epochs is $sk$. Let $X_i$, $0 \leq i \leq (sk-1)$, be a random variable defined to be the number of trials in the $i$th epoch, so that

$$X = \sum_{i=0}^{sk-1} X_i \ .$$

Let $p_i$ be the probability of success on any trial of the $i$th epoch. A sample drawn in the $i$th epoch is not good if the draw is made from a partition from which $s$ vertices have already been drawn. In the $i$th epoch there can be at most $\lfloor \frac{i}{s} \rfloor$ partitions from which $s$ samples have been already drawn. Hence, there are still at least $(k - \lfloor \frac{i}{s} \rfloor)$ partitions from which good samples can be drawn. Then, since the probability of getting a sample from any particular partition is at least $\frac{1}{l}$, the probability of the sample being good is

$$p_i \geq \sum_{j=1}^{k-\lfloor \frac{i}{s} \rfloor} \frac{1}{l} = \frac{k - \lfloor \frac{i}{s} \rfloor}{l} \ .$$

The random variable $X_i$ is geometrically distributed with parameter $p_i$ and hence

$$E[X_i] = \frac{1}{p_i} \leq \frac{l}{k - \lfloor \frac{i}{s} \rfloor} \ .$$

Therefore, by linearity of expectation,

$$\begin{aligned} E[X] &= E[\sum_{i=0}^{sk-1} X_i] = \sum_{i=0}^{sk-1} E[X_i] \\ &\leq \sum_{i=0}^{sk-1} \frac{l}{k - \lfloor \frac{i}{s} \rfloor} = l \sum_{j=1}^{k} \sum_{h=0}^{s-1} \frac{1}{k - \lfloor \frac{(j-1)s+h}{s} \rfloor} \\ &= sl \sum_{j=1}^{k} \frac{1}{k-(j-1)} = sl \sum_{i=1}^{k} \frac{1}{i} \\ &\approx sl \ln k + O(sl) \ . \end{aligned}$$

That completes the proof. ∎

Using this lemma, we present the following result that shows that if we draw $n = csl \ln k \approx cE[X]$ samples from $\mathcal{X}$, where $c$ is an appropriately chosen constant, then we we will get at least $s$ samples from each optimal partition with high probability.

**Lemma 6** *If $csl \ln k$ samples are drawn uniformly at random from $\mathcal{X}$ the probability of getting at least $s \gg 1$ samples from each of the optimal partitions is more than $(1 - \frac{1}{k^d})$, where $c$ and $d$ are constants such that $c \geq \frac{1}{\ln k}$ and $d \leq \frac{s}{\ln k}\{c \ln k - \ln(4c \ln k)\} - 1$.*

*Proof:* Let $E_h^n$ denote the event that less than $s$ points have been sampled from the partition $S_h^*$ in the first $n$ draws. Let $q_h$ be the probability that a uniformly random sample is chosen from $S_h^*$. Then, $q_h \geq \frac{1}{l}$. We shall consider the number of

draws $n = csl \ln k$ for a constant $c \geq \frac{1}{\ln k}$. Then,

$$P[E_h^n] = \sum_{j=0}^{s-1} \binom{n}{j} q_h^j (1-q_h)^{n-j} \leq \sum_{j=0}^{s-1} \binom{n}{j} \left(\frac{1}{l}\right)^j \left(1-\frac{1}{l}\right)^{n-j} .$$

Now, since the expectation for the binomial distribution $B(csl \ln k, \frac{1}{l})$ is $cs \ln k \geq s > (s-1)$, the largest term in the summation is the term corresponding to $i = s-1$. Hence,

$$P[E_i^n] \leq s\binom{n}{s-1} \left(\frac{1}{l}\right)^{s-1} \left(1-\frac{1}{l}\right)^{n-s+1} \leq s\binom{n}{s} \left(\frac{1}{l}\right)^{s} \left(1-\frac{1}{l}\right)^{n-s} .$$

Since $n\left(\frac{1}{l}\right)\left(\frac{l-1}{l}\right) \approx \frac{n}{l} = cs \ln k \geq s \gg 1$, applying the *Poison theorem* [Pap91],

$$\binom{n}{s} \left(\frac{1}{l}\right)^{s} \left(1-\frac{1}{l}\right)^{n-s} \approx e^{-\frac{n}{l}} \frac{\left(\frac{n}{l}\right)^s}{s!} .$$

Hence,

$$P[E_i^n] \leq se^{-\frac{n}{l}} \frac{\left(\frac{n}{l}\right)^s}{s!} .$$

Using the fact that the probability of a union of events is always less than the sum of the probabilities of these events, we obtain

$$P[X > n] = P[\cup_{i=1}^{k} E_i^n] \leq \sum_{i=1}^{k} P[E_i^n] \leq kse^{-\frac{n}{l}} \frac{\left(\frac{n}{l}\right)^s}{s!} .$$

Putting $n = csl \ln k$, we get

$$\begin{aligned} P[X > n] &\leq kse^{-\frac{n}{l}} \frac{\left(\frac{n}{l}\right)^s}{s!} = kse^{-cs \ln k} \frac{(cs \ln k)^s}{s!} \\ &\leq (ce)^s sk^{1-cs} (\ln k)^s \qquad (\text{since } s! \geq \left(\frac{s}{e}\right)^s) \\ &= \left(cek^{\frac{1}{s}-c} \ln k s^{\frac{1}{s}}\right)^s . \end{aligned}$$

Then $P[X \leq n] \geq (1 - \frac{1}{k^d})$, or, $P[X > n] \leq \frac{1}{k^d}$ if $\left(cek^{\frac{1}{s}-c} \ln k s^{\frac{1}{s}}\right)^s \leq \frac{1}{k^d}$, which is true if $k^{c-\frac{d+1}{s}} \geq ces^{\frac{1}{s}} \ln k$. Now, $s^{\frac{1}{s}} \leq e^{\frac{1}{e}}$, $\forall s$, and hence $es^{\frac{1}{s}} \leq ee^{\frac{1}{e}} < 4$. So, the probability bound is satisfied if $k^{c-\frac{d+1}{s}} \geq 4c \ln k$. A simple algebraic manipulation of the last inequality gives our desired upper bound on $d$. ■

To help understand the result, consider the case where $l = k = 10$ and we want at least $s = 50$ points from each of the partitions. Table 6.1 shows the total number of points that need to be sampled for different levels of confidence. Note that if

| d | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Confidence, $100(1 - \frac{1}{k^d})\%$ | 90.000 | 99.000 | 99.900 | 99.990 | 99.999 |
| Number of Samples, n | 1160 | 1200 | 1239 | 1277 | 1315 |

Table 6.1: Number of samples required to achieve a given confidence level for k=10 and s=50

the $k$ optimal partitions are of equal size and $csk \ln k$ points are sampled uniformly at random, the expected number of points from each partition is $cs \ln k$. Thus the underlying structure is expected to be preserved in this smaller sampled set and the chances of wide variations from this behavior is very small. For example, for the 99.99% confidence level in Table 6.1, the average number of samples per partition is 127, which is only 2.5 times the minimum sample size that is desired.

## 6.3 Clustering of the Sampled Set

The second step involves clustering the set of $n$ sampled points, $\mathcal{X}_s$. The only requirement from this stage is to obtain a $k$-partitioning of $\mathcal{X}_s$ and have a representative $\boldsymbol{\mu}_h, h = 1, \cdots, k$ corresponding to each partition. There are several clustering formulations that satisfy this requirement, e.g., clustering using Bregman divergences [BMDG03] for which the optimal representatives are given by the centroids, clustering using cosine-type similarities [DM01, BDGS03] for which the optimal rep-

resentatives are given by the $L_2$-normalized centroids, convex clustering [MS03] for which the optimal representatives are given by generalized centroids, etc. Since there are already a large number of clustering formulations that satisfy our requirement and the main issue we are trying to address is how to make the existing clustering algorithms scalable, we do not propose any new algorithm for clustering the sampled set. Instead, we just review two widely used existing algorithms, viz Euclidean kmeans clustering [Mac67] and spherical kmeans clustering [DM01], for which experimental results will be presented.

### 6.3.1 Euclidean kmeans

Euclidean kmeans takes the set of $n$ points to be clustered, $\mathcal{X}_s$, and the number of clusters, $k$, as an input. The Euclidean kmeans problem [Mac67, DHS00] is to get a $k$-partitioning $\{S_h\}_{h=1}^k$ of $\mathcal{X}_s$ such that the following objective function,

$$\mathcal{J}_{\text{kmeans}} = \frac{1}{n} \sum_{h=1}^{k} \sum_{\mathbf{x} \in S_h} (\mathbf{x} - \mu_h)^2,$$

is minimized. The `kmeans` algorithm gives a simple greedy solution to this problem that guarantees a local minimum of the objective function. The algorithm starts with a random guess for $\{\mu_h\}_{h=1}^k$. At every iteration, each point $x$ is assigned to the partition $S_{h^*}$ corresponding to its nearest centroid $\mu_{h^*}$. After assigning all the points, the centroids of each partition is computed. These two steps are repeated till convergence. It can be shown that the iterations converge in a finite number of iterations to a local minimum of the objective function. The reason we chose the `kmeans` algorithm is its wide usage. Further, since `kmeans` is a special case of Bregman clustering algorithms [BMDG03] as well as convex clustering algorithms [MS03], the experiments of section 6.5 gives some intuition as to how the proposed framework will perform in more general settings.

### 6.3.2 Spherical kmeans

Spherical kmeans [DM01] is applicable to data points on the surface of the unit hypersphere and has been successfully applied and extended to clustering of natural high-dimensional datasets [BDGS03]. As discussed in Chapter 5, the spherical kmeans problem is to get a $k$-partitioning $\{S_h\}_{h=1}^k$ of a set of $n$ data points $\mathcal{X}_s$ on the unit hypersphere such that the following objective function,

$$\mathcal{J}_{\text{spkmeans}} = \frac{1}{n} \sum_{h=1}^{k} \sum_{\mathbf{x} \in S_h} \mathbf{x}^T \mu_h,$$

is maximized. The `spkmeans` algorithm [DM01] gives a simple greedy solution to this problem that guarantees a local maximum of the objective function. Like `kmeans`, `spkmeans` starts with a random guess for $\{\mu_h\}_{h=1}^k$. At every iteration, each point $x$ is assigned to the partition $S_{h^*}$ corresponding to its most similar centroid $\mu_{h^*}$, and finally, the centroids of each partition are computed. It can be shown that the algorithms converge in a finite number of iterations to a local maximum of the objective function. Successful application of this algorithm to natural datasets, e.g., text, gene-expression data, etc., gives us the motivation to study the algorithm in greater detail.

## 6.4 Populating and Refining the Clusters

After clustering the $n$ points sampled from the original data $\mathcal{X}$, the remaining $(N-n)$ points can be assigned to the clusters, satisfying the balancing constraint. Subsequently, these clusters need to be refined to get the final partitioning. This is the final and perhaps most critical step since it has to satisfy the balancing requirements while maintaining the scalability of the overall approach. In fact, the previous two steps can essentially be considered as a good way of finding an initialization for

an iterative refinement algorithm that works with $\mathcal{X}$ and minimizes the objective function $\mathbb{L}$. In this section, we discuss a novel scheme for populating the clusters and then iteratively refining the clustering.

### 6.4.1 Overview

There two parts in the proposed scheme:

1. Populate: First, the points that were not sampled, and hence do not currently belong to any cluster, are assigned to the existing clusters in a manner that satisfies the balancing constraints while ensuring good quality clusters.

2. Refine: Iterative refinements are done to improve on the clustering objective function while satisfying the balancing constraints all along.

Hence, part 1 gives a reasonably good feasible solution, i.e., a clustering in which the balancing constraints are satisfied. Part 2 iteratively refines the solution while always remaining in the feasible space.

Let $n_h$ be the number of points in cluster $h$, so that $\sum_{h=1}^{k} n_h = n$. Let $\mathcal{X}_u = \{\mathbf{x}_1, \cdots, \mathbf{x}_{N-n}\}$ be the set of $(N-n)$ non-sampled points. The final clustering needs to have at least $m$ points per cluster to be feasible. Let $b = \frac{mk}{N}$, where $0 \leq b \leq 1$ since $m \leq N/k$, be the *balancing fraction*. For any assignment of the members of $\mathcal{X}$ to the clusters, let $\ell_i \in \{1, \ldots, k\}$ denote the cluster assignment of $\mathbf{x}_i$. Further, let $S_h = \{\mathbf{x}_i \in \mathcal{X} | \ell_i = h\}$.

#### Part 1: Populate

In the first part, we just want a reasonably good feasible solution so that $|S_h| \geq m, \forall h$. Hence, since there are already $n_h$ points in $S_h$, we need to assign $[m - n_h]_+$ more points to $S_h$, where $[x]_+ = \max(x, 0)$. Ideally, each point in $\mathcal{X}_u$ should be assigned to the nearest cluster so that $\forall \mathbf{x}_i,\ d(\mathbf{x}_i, \mu_{\ell_i}) \leq d(\mathbf{x}_i, \mu_h), \forall h$. Such

assignments will be called *greedy* assignments. However, this need not satisfy the balancing constraint. So, we do the assignment of all the points as follows:

(1) First, exactly $[m - n_h]_+$ points are assigned to cluster $h, \forall h$, such that for each $\mathbf{x}_i$ that has been assigned to a cluster,

**either** it has been assigned to its nearest cluster, i.e.,

$$d(\mathbf{x}_i, \mu_{\ell_i}) \leq d(\mathbf{x}_i, \mu_h), \quad \forall h \ ,$$

**or** all clusters $h'$ whose representatives $\mu_{h'}$ are nearer to $\mathbf{x}_i$ than its own representative $\mu_{\ell_i}$ already have the required number of points $[m - n_{h'}]_+$, all of which are nearer to $\mu_{h'}$ than $\mathbf{x}_i$, i.e.,

$$\forall h' \text{ such that } d(\mathbf{x}_i, \mu_{\ell_i}) > d(\mathbf{x}_i, \mu_{h'}), \qquad d(\mathbf{x}_i, \mu_{h'}) \geq d(\mathbf{x}', \mu_{h'}), \ \forall \mathbf{x}' \in S_{h'} \ .$$

(2) The remaining points are greedily assigned to their nearest clusters.

Condition (1) is motivated by the stable marriage problem that tries to get a stable match of $n$ men and $n$ women, each with his/her preference list for marriage over the other set [GI89]. The populate step can be viewed as a generalization of the standard stable marriage setting in that there are $k$ clusters that want to "get married", and cluster $h$ wants to "marry" at least $[m - n_h]_+$ points. Hence, an assignment of points to clusters that satisfies condition (1) will be called a *stable* assignment and the resulting clustering is called *stable*.

**Part 2: Refine**

In the second part, starting from the clustering obtained from the first part, feasible iterative refinements are done until convergence. Note that at this stage, each point $\mathbf{x}_i \in \mathcal{X}$ is in one of the clusters and the balancing constraints are satisfied, i.e.,

$|S_h| \geq m, \forall h$. There are two ways in which a refinement can be done, and we iterate between these two steps and the updation of the cluster representative:

(1) If a point $\mathbf{x}_i$ can be moved, without violating the balancing constraint, to a cluster whose representative is nearer than its current representative, then the point can be safely re-assigned. First, all such possible individual re-assignments are done.

(2) Once all possible individual re-assignments are done, there may still be groups of points in different clusters that can be simultaneously re-assigned to reduce the cost without violating the constraints. In this stage, all such possible group re-assignments are done.

After all the individual and group reassignments are made, the cluster representatives are re-estimated. Using the re-estimated representatives, a new set of re-assignments are possible and the above two steps are performed again. The process is repeated until no further updates can be done, and the refinement algorithm terminates.

### 6.4.2 Details of Part 1: Populate

In this subsection, we discuss `poly-stable`, an algorithm to make stable assignment of $[m - n_h]_+$ points per cluster. The algorithm is presented in detail in Algorithm 8. First the distance $d(\mathbf{x}, \mu_h)$ between every unassigned point $\mathbf{x} \in \mathcal{X}_u$ and every cluster representative is computed. For each cluster $S_h$, all the non-sampled $(N - n)$ points are sorted in increasing order of their distance with $\mu_h, h = 1, \ldots, k$. Let the ordered list of points for cluster $S_h$ be denoted by $\Pi_h$. Let $m_h$ denote the number of points yet to be assigned to cluster $S_h$ at any stage of the algorithm to satisfy the constraints. Initially $m_h = [m - n_h]_+$. The algorithm terminates when $m_h = 0, \forall h$.

The basic idea behind the algorithm is "cluster proposes, point disposes". In every iteration, each cluster $S_h$ proposes to its nearest $m_h$ points. Every point that

has been proposed, gets temporarily assigned to the nearest among its proposing clusters and rejects any other proposals. Note that if a point has received only one proposal, it gets temporarily assigned to the only cluster which proposed to it and there is no rejection involved. For each cluster, $m_h$ is recomputed. If $m_h \neq 0$, cluster $S_h$ proposes the next $m_h$ points from its sorted list $\Pi_h$ that have not already rejected its proposal. Each of the proposed points accepts the proposal either if it is currently unassigned or if the proposing cluster is nearer than the cluster $S_{h'}$ to which it is currently assigned. In the later case, the point rejects its old cluster $S_{h'}$ and the cluster $S_{h'}$ loses a point so that $m_{h'}$ goes up by 1. This process is repeated until $m_h = 0, \forall h$ and the algorithm terminates.

Now we show that this algorithm indeed satisfies all the required conditions and hence ends up in a stable assignment. Before giving the actual proof, we take a closer look at what exactly happens when an assignment is unstable. Let $(\mathbf{x}_i \rightarrow S_h)$ denote the fact that the point $\mathbf{x}_i$ has been assigned to the cluster $S_h$. An assignment is unstable if there exist at least two assignments $(\mathbf{x}_{i_1} \rightarrow S_{h_1})$ and $(\mathbf{x}_{i_2} \rightarrow S_{h_2})$, $h_1 \neq h_2$, such that $\mathbf{x}_{i_1}$ is nearer to $\mu_{h_2}$ than its own cluster representative $\mu_{h_1}$ and $\mu_{h_2}$ is nearer to $\mathbf{x}_{i_1}$ than $\mathbf{x}_{i_2}$. The point-cluster pair $(\mathbf{x}_{i_1}, S_{h_2})$ is said to be dissatisfied under such an assignment. An assignment in which there are no dissatisfied point-cluster pairs is a stable assignment and satisfies the constraints. Next, we show that there will no dissatisfied point-cluster pair after the algorithm terminates.

**Lemma 7** `poly-stable` *gives a stable assignment.*

*Proof:* If possible, let `poly-stable` give an unstable assignment. Then there must exist two assignments $(\mathbf{x}_{i_1} \rightarrow S_{h_1})$ and $(\mathbf{x}_{i_2} \rightarrow S_{h_2})$ such that $(\mathbf{x}_{i_1}, S_{h_2})$ is a dissatisfied point-cluster pair. Then, since $\mathbf{x}_{i_1}$ is nearer to $S_{h_2}$ than $\mathbf{x}_{i_2}$, $S_{h_2}$ must have proposed to $\mathbf{x}_{i_1}$ before $\mathbf{x}_{i_2}$. Since $\mathbf{x}_{i_1}$ is not assigned to $S_{h_2}$, $\mathbf{x}_{i_1}$ must have either rejected the proposal of $S_{h_2}$ meaning it was currently assigned to a cluster which was nearer than $S_{h_2}$, or accepted the proposal initially only to reject it later meaning

**Algorithm 8** `poly-stable`

**Input:** The existing clusters $\{S_h\}_{h=1}^k$, the cluster representatives $M = \{\mu_h\}_{h=1}^k$, the required minimum cluster size $m$, and the set of unassigned data points $\mathcal{X}_u$.
**Output:** A disjoint stable $k$-partitioning $\{S_h\}_{h=1}^k$ of $\mathcal{X}$ such that $|S_h| \geq m, \forall h$

**Method:**

Mark all $\mathbf{x} \in \mathcal{X}_u$ as *free*
**for** $h = 1$ to $k$ **do**
  $\Pi_h \leftarrow$ Sort $\mathcal{X}_u$ in increasing order according to $d(\mathbf{x}, \mu_h), \mathbf{x} \in \mathcal{X}_u, \mu_h \in M$
  $m_h \leftarrow [m - |S_h|]_+$ {number of points to be assigned}
  $\iota_h \leftarrow 1$ {index in the sorted list $\Pi_h$}
**end for**
$\lambda \leftarrow \sum_{h=1}^k m_h$
**while** $\lambda > 0$ **do**
  **for** $h = 1$ to $k$ **do**
    **if** $m_h > 0$ **then**
      $m_h^* \leftarrow m_h$
      { Cluster $C_h$ proposes to the *next* $m_h$ points $\Pi_h(i), i = \iota_h, \cdots, (\iota_h + m_h - 1)$ }
      **for** $i = \iota_h$ to $(\iota_h + m_h - 1)$ **do**
        **if** $\Pi_h(i)$ is *free* **then**
          $S_h \leftarrow S_h \cup \{\Pi_h(i)\}$
          Mark $\Pi_h(i) \in \mathcal{X}^{(u)}$ as *engaged to* $h$
          $m_h \leftarrow m_h - 1$
        **else if** $\Pi_h(i)$ is *engaged to* $h'$ but $d(v, \mu_h) < d(v, \mu_{h'})$ **then**
          $S_h \leftarrow S_h \cup \{\Pi_h(i)\}$, $S_{h'} \leftarrow S_{h'} \setminus \{\Pi_h(i)\}$
          Mark $\Pi_h(i) \in \mathcal{X}_u$ as *engaged to* $h$
          $m_h \leftarrow m_h - 1$, $m_{h'} \leftarrow m_{h'} + 1$
        **end if**
      **end for**
      $\iota_h \leftarrow \iota_h + m_h^*$
    **end if**
  **end for**
  $\lambda \leftarrow \sum_{h=1}^k m_h$
**end while**

it got a proposal from a cluster nearer than $S_{h_2}$. Since a point only improves its assignments over time, the cluster $S_{h_1}$ to which $\mathbf{x}_{i_1}$ is finally assigned must be nearer to it than the cluster $S_{h_2}$ which it rejected. Hence $\mathbf{x}_{i_1}$ cannot be dissatisfied which contradicts the initial assumption. So, the final assignment is indeed stable. ■

Next we look at the total number of proposals that are made before the algorithm terminates. After a cluster proposes to a point for the first time, there are three possible ways this particular point-cluster pair can behave during the rest of the algorithm: (i) the point immediately rejects the proposal in which case the cluster never proposes to it again; (ii) the point accepts it temporarily and rejects it later — the cluster obviously does not propose to it during the acceptance period and never proposes to it after the rejection; (iii) the point accepts the proposal and stays in that cluster till the algorithm terminates — obviously the cluster does not propose to it again during this time. Hence, each cluster proposes each point at most once and so the maximum number of proposals possible is $k \times (bN - n)$.

We take a look at the complexity of the proposed scheme. Following the above discussion, the complexity from the proposal part is $O(k(bN - n))$. Before starting the proposals, the sorted lists of distances of all the points from each of the clusters have to be computed, which has a complexity of $O(k(N - n)\log(N - n))$. And after `poly-stable` terminates, the remaining $N(1 - b)$ points are greedily assigned to their nearest clusters. Note that the greedy assignments at this stage does not hamper the feasibility or the stability of the resulting clustering. So, the total complexity of the first part of the proposed scheme is $O(k(N - n)(\log(N - n) + k(bN - n) + N(1 - b)) = O(kN \log N)$.

After all the points are assigned to clusters using `poly-stable`, the cluster representatives $\{\mu_h\}_{h=1}^{k}$ are updated such that for $h = 1, \ldots, k$,

$$\mu_h = \operatorname*{argmin}_{\mu} \sum_{\mathbf{x} \in S_h} d(\mathbf{x}, \mu) \ .$$

We assume that there is an efficient way to get the representatives. Note that each new representative will be at least as good as the old representatives in terms of the cost function. Hence, optimal updation of the cluster representatives results in a decrease in the cost function value.

### 6.4.3 Details of Part 2: Refine

In this subsection, we discuss `refine`, an algorithm to do iterative refinements in order to get a better solution to the clustering problem while satisfying the constraints all along. Note that this part of the scheme applies to all points in $\mathcal{X}$, and does not differentiate between a sampled or non-sampled point. Further, the input to this part is an existing disjoint partitioning $\{S_h\}_{h=1}^{k}$ of $\mathcal{X}$ such that $|S_h| = n_h \geq m, \forall h$. As outlined earlier, `refine` has two parts in every iteration: the first part involves performing *individual re-assignments*, where individual points are re-assigned with a guaranteed decrease in the objective while satisfying constraints; the second part involves performing *group reassignments* with the same guarantees.

#### Individual Re-assignments

The individual re-assignment (IR) runs over all the points $\mathbf{x} \in \mathcal{X}$. If, for a point $\mathbf{x}_i$ in cluster $S_{\ell_i}$,

(a) there is a cluster $h$ whose representative $\boldsymbol{\mu}_h$ is nearer to $\mathbf{x}_i$ than its current representative $\boldsymbol{\mu}_{\ell_i}$, i.e.,

$$\exists h \neq \ell_i, \qquad d(\mathbf{x}_i, \boldsymbol{\mu}_{\ell_i}) > d(\mathbf{x}_i, \boldsymbol{\mu}_h) \ ,$$

(b) cluster $S_{\ell_i}$ can afford to let go a point without violating the constraints, i.e., $n_{\ell_i} > m$,

then $\mathbf{x}_i$ can be safely reassigned to (any such) cluster $h$. Clearly, the overall objective function decreases without violating any constraints. To obtain the maximum decrease, $\mathbf{x}_i$ should be assigned to its nearest cluster. After a pass through the entire data, for any point $\mathbf{x}_i$

**either** it is in its nearest cluster, and hence there is no reason to re-assign it,

**or** it is not in its nearest cluster, but it cannot be re-assigned since that will violate the balancing constraints for the cluster it is currently in.

For the first set of points, no updates are necessary. For the second set, we investigate if group re-assignments can improve the objective while satisfying constraints.

### Group Re-assignments

For every point $\mathbf{x} \in \mathcal{X}$, let $H^{(\mathbf{x})}$ be the set of clusters $h$ whose cluster representatives $\mu_h$ are nearer to $\mathbf{x}$ than its current cluster representative $\boldsymbol{\mu}_{\ell_i}$, i.e.,

$$H^{(\mathbf{x})} = \{h | d(\mathbf{x}, \mu_{\ell_i}) > d(\mathbf{x}, \mu_h)\} \ ,$$

Clearly, for points already in their nearest cluster, $H^{(\mathbf{x})}$ is the null set. Using the sets $H^{(\mathbf{x})}$, $\forall \mathbf{x} \in \mathcal{X}$, we construct a $k$-vertex potential assignment graph $G = (V, E)$ with $V = \{h\}_{h=1}^{k}$, one vertex corresponding to every cluster, such that for every point $\mathbf{x}_i \in \mathcal{X}$, there is a directed edge of unit capacity from vertex $\ell_i$ to all vertices $h \in H^{(\mathbf{x}_i)}$. The total capacity on every directed edge is consolidated to get a single integer value, and edges with zero capacity are considered non-existent. Note that a cycle in this integer weighted directed graph, suggests a set of group re-assignments of points that is guaranteed to decrease the clustering objective function. In order to do group re-assignments, the algorithm finds the strongly connected components of $G$ and keeps removing weighted cycles from each component till there are no cycles in the graph.

The final step is to re-estimate the representatives of the individual clusters so that the clustering objective function is further decreased. Note that the re-estimation of the representatives should be followed by individual and group re-assignments, and the process is to be repeated till convergence. The details of the algorithm `refine` are presented in Algorithm 9.

## 6.5 Experimental Results

In this section, we first briefly discuss the complexity of the proposed framework. Then, we present empirical results on several high-dimensional text datasets using different performance evaluation criterion for comparison.

The sampling of the $n$ points out of $N$ uniformly at random has a complexity of $O(N)$. If the clustering algorithm on the sampled data is a variant of `KMeans`, the computational complexity is $O(t_0 kn)$ where $t_0$ is the number of iterations. As shown earlier, the `poly-stable` algorithm has a complexity of $O(kN \log N)$. The individual re-assignments of the refinement step is $O(N)$, whereas the group re-assignments take $O(kN + t_j k^2)$ where $t_j$ is the number of strongly connected components in the $j^{th}$ iteration. If the refinement is performed for $T$ iterations, then the complexity of the step is $O(TN + TkN + \sum_{j=1}^{T} t_j k^2) = O(kN)$. So, the total complexity of the scheme is $O(kN \log N)$, assuming $t_0, T, \{t_j\}_{j=1}^{T}$ are constants. Note that this is better than the $\Omega(N^2)$ complexity of graph-based algorithms that can provide balancing. It is worse than the $O(kN)$ complexity (assuming constant number of iterations) of the iterative greedy relocation algorithms such as `KMeans`, but such algorithms are not guaranteed to satisfy the balancing constraints.

### 6.5.1 Datasets

The datasets that we used for empirical validation and comparison of our algorithms were carefully selected to represent some typical clustering problems. We

**Algorithm 9** `refine`

**Input:** A disjoint stable $k$-partitioning $\{S_h\}_{h=1}^k$ of $\mathcal{X}$ such that $|S_h| \geq m$; a set of cluster representatives $\mu_h$

**Output:** A disjoint $k$-partitioning $\{S_h^*\}_{h=1}^k$ of $\mathcal{X}$ and a set of representatives $\{\boldsymbol{\mu}_h^*\}_{h=1}^k$ such that $|S_h^*| \geq m$, $L(\{\mu_h^*, S_h^*\}_{h=1}^k) \leq L(\{\mu_h, S_h\}_{h=1}^k)$ and $\{\mu_h^*, S_h^*\}_{h=1}^k$ is a local minimum of $\mathbb{L}(\cdot)$.

**Method:**

**repeat**
  $\theta \leftarrow 0$
  {Update the cluster means}
  **for** $h = 1$ to $k$ **do**
    $\mu_h \leftarrow \frac{1}{|S_h|} \sum_{x \in S_h} x$
  **end for**
  {Reassign points}
  **for all** $\mathbf{x}_i \in \mathcal{X}$ **do**
    $H^{(\mathbf{x}_i)} \leftarrow \emptyset$
    {Individual Reassignments}
    **for** $h = 1$ to $k$ **do**
      **if** $d(x, \mu_h) < d(x, \mu_{h^*})$ **then**
        **if** $|S_{\ell_i}| > m$ **then**
          $S_h \leftarrow S_h \cup \{x\},\ S_{\ell_i} \leftarrow S_{\ell_i} \setminus \{x\},\ \ell_i \leftarrow h,\ \theta \leftarrow \theta + 1$
        **else**
          $H^{(\mathbf{x}_i)} \leftarrow H^{(\mathbf{x}_i)} \cup \{h\}$
        **end if**
      **end if**
    **end for**
  **end for**
  {Group Reassignments}
  Construct potential assignment graph $G = (V, E)$ from $\{H^{(\mathbf{x})}\}_{\mathbf{x} \in \mathcal{X}}$
  $C \leftarrow$ `strongly-connected-components`$(G)$
  **for all** $\gamma \in C$ **do**
    **while** there exists *back-edge* in `DFS`$(\gamma)$ **do**
      $\lambda \leftarrow$ the cycle corresponding to this back-edge
      $\theta \leftarrow \theta +$ minimum edge-weight in $\lambda$
      $\lambda_\theta \leftarrow \lambda$ with all edge-weights set to $\theta$
      reassign points according to directed cycle $\lambda_\theta$
      $\gamma \leftarrow \gamma \setminus \lambda_\theta$
    **end while**
  **end for**
**until** $\theta = 0$
$\{S_h^*\}_{h=1}^k \leftarrow \{S_h\}_{h=1}^k$

also created various subsets of some of the datasets for gaining greater insight into the nature of clusters discovered or to model some particular clustering scenario (e.g. balanced clusters, skewed clusters, overlapping clusters etc.). Although some of the datasets were described in Chapter 5, we discuss all the datasets here for completeness.

- classic3: Classic3 is a well known collection of documents used for text analysis. It is an easy dataset to cluster since it contains documents from three well-separated sources. Moreover, the intrinsic clusters are largely balanced. The corpus contains 3893 documents, among which 1400 CRANFIELD documents are from aeronautical system papers, 1033 MEDLINE documents are from medical journals, and 1460 CISI documents are from information retrieval papers. The particular vector space model used had a total of 4666 features (words). Thus, each document is represented as a vector in a 4666-dimensional space. The dataset serves as a sanity check for the algorithms.

- news20: The CMU Newsgroup dataset, hereafter called news20, is a widely used compilation of documents. We tested our algorithms on not only the original dataset, but on a variety of subsets with differing characteristics to explore and understand the behavior of our algorithms. The standard dataset is a collection of 19,997 messages, gathered from 20 different USENET newsgroups. One thousand messages are drawn from the first 19 newsgroups, and 997 from the twentieth. The headers for each of the messages are then removed to avoid biasing the results. The particular vector space model used had 25,924 words. The dataset embodies the features characteristic of a typical text dataset—high-dimensionality, sparsity and some significantly overlapping clusters.

- small-news20: We formed this subset of news20 by sampling 2000 messages from original dataset. We sampled 100 messages from each category in the

original dataset. Hence the dataset has balanced clusters (though there may be overlap). The dimensionality of the data was 13,406. Since the dimensionality of the data is much smaller than the number of data points, this is a harder dataset to cluster.

- similar-1000: We formed another subset of news20 by choosing 3 somewhat similar newsgroups: talk.politics.guns, talk.politics.mideast, talk.politics.misc. The dataset has a total of 3000 documents, with 1000 documents per newsgroup, and a dimensionality of 10,083. The dataset has a fair amount of overlaps in high dimensions and is hence a difficult dataset to cluster.

- yahoo: The Yahoo20 dataset (K-series) is a collection of 2340 Yahoo news articles belonging one of 20 different Yahoo categories, and has a dimensionality of 24273. The salient feature of this dataset is that the natural clusters are not at all balanced, with cluster sizes ranging from 9 to 494. This is where it significantly differs from the previous datasets. This dataset helps in studying the effect of forcing balanced clustering in a naturally unbalanced data.

- nsf-top30: The NSF dataset contains abstracts of all NSF proposals that were granted funding over a period of 14 years from 1990-2003. The nsf-top30 is a mildly pruned version of the data that contains the top 30 categories, in terms of number of grants, that were funded. The data consists of 128,111 abstracts from 30 categories with sizes ranging from 9911 to 1447. In fact, all categories with more than 1000 grants were selected. The data has a dimensionality of 45,998. This is a rather large and quite sparse dataset since abstracts are typically much smaller than normal documents.

### 6.5.2 Algorithms

We compared 6 different algorithms on every dataset.

- `KMeans` is the widely used and popular iterative relocation clustering algorithm. There is one important modification we make in the way the algorithm was run — we ran the data on the $L_2$ normalized version of the data. There are two reasons for this: (i) all algorithms run on the same representation of the data, and (ii) it has been well established that applying kmeans on the original sparse high-dimensional data gives poor results [DM01].

- `SPKMeans` is the spherical kmeans algorithm [DM01] outlined in section 6.3. Motivated by research in information retrieval, the algorithm uses cosine similarity between data points and cluster representatives and has been shown to give good results on several benchmark text datasets [DM01, BDGS03].

  Note that both the above algorithms do not have any way to ensure balanced clustering. The next algorithm uses all the components of the proposed framework, and hence guarantees balanced clusterings with a scalable algorithm.

- `SPKpr` uses `SPKMeans` as the base clustering algorithm and uses both the populate (`p`) and refine (`r`) steps as outlined in section 6.4. This algorithm is guaranteed to satisfy any balancing constraints given as input, although the quality of clusters may deteriorate under stringent conditions.

  We also perform lesion studies using three other algorithms, in which one or more components of the proposed framework are missing.

- `SPKpnr` uses `SPKMeans` as the base clustering algorithm. It uses the populate (`p`) step, but does not refine (`nr`) the resulting clusters. The algorithm satisfies any given balancing constraints but need not give good results since the feasible solution is not refined.

- `SPKgpnr` also uses `SPKMeans` for clustering. It uses a greedy populate (`gp`) scheme where every point is assigned to the nearest cluster. Further, no re-

finements (`nr`) are done after the greedy populate step. Clearly, this algorithm is not guaranteed to satisfy balancing constraints.

- `SPKgpr` uses `SPKmeans` as the base clustering algorithm. It uses greedy populate (`gp`) to put points into clusters, but performs a full refinement (`r`) after that. The algorithm is not guaranteed to satisfy the balancing constraints since the populate step is greedy and the refinements do not start from a feasible clustering.

In a tabular form, the four algorithms can be presented as follows:

| | No Refine | Refine | Balancing |
|---|---|---|---|
| Greedy Populate | `SPKgpnr` | `SPKgpr` | No Guarantee |
| Populate | `SPKpnr` | `SPKpr` | Guaranteed |

### 6.5.3 Methodology

Performance of the algorithms are evaluated using one measure for the quality of clustering and two measures for the balancing of cluster sizes. The measure for cluster quality evaluates how the clustering agrees with the hidden true labels of the data. The measures for cluster sizes evaluate how well balanced the cluster sizes are in the average as well as worst case.

Quality of clustering is evaluated using the following measure:

- Normalized Mutual Information (NMI) is used to measure the agreement of the assigned cluster labels and the true class labels from the confusion matrix of the assignments. Mutual information (MI) gives the amount of statistical similarity between the cluster and class labels [CT91]. If $X$ is a random variable for the cluster assignments and $Y$ is a random variable for the true labels on the same data, then their MI is given by $I(X;Y) = E[\ln \frac{p(X,Y)}{p(X)p(Y)}]$ where the expectation is computed over the joint distribution of $(X, Y)$ estimated from the particular clustering of the dataset under consideration. In order to get a number

in $[0, 1]$, we normalize the MI to get $NMI(X;Y) = I(X;Y)/\sqrt{H(X)H(Y)}$, where $H(X), H(Y)$ are the entropies of the marginal distributions of $X, Y$ respectively.

Note that the above normalization is different from the one used in Chapter 5. There is no consensus on the best normalization, or, for that matter, the best external measure to evaluate clustering. We will discuss this issue in Chapter 7.

Quality of balancing is evaluated using the following measures:

- Standard Deviation in cluster sizes is the first and perhaps most obvious way to evaluate balancing. For algorithms that are guaranteed to give balanced clusters, the standard deviation becomes smaller as the balancing fraction is increased.

- The ratio between the minimum to average cluster sizes is our second measure of evaluation. For $N$ data points put into $k$ clusters, the average cluster size is just $N/k$. For a given clustering, we compare the size of the smallest cluster to this average.

All results reported here have been averaged over 10 runs. All algorithms were started with the same random initialization to ensure fairness of comparison. Each run was started with a *different* random initialization. However, no algorithm was restarted within a given run and all of them were allowed to run to completion. Since the standard deviations over 10 runs were reasonably small, to reduce clutter, we have chosen to omit a display of error bars in our plots.

### 6.5.4 Results

Experiments were run for particular choices of datasets, algorithms, number of samples and balancing fraction. Note that the choice of number of samples and the balancing fraction may be in conflict in the sense that both cannot be satisfied at

the same time. For example, for a 3-clustering task with 10,000 points, if the number of samples is 5,000 and the balancing fraction is 0.9, then, if the preliminary 3-clustering of 5,000 points puts all points in one cluster with two other empty clusters, it is not possible to achieve a balancing fraction of 0.9 in the final clustering, i.e., at least 3000 points per cluster. In order to prevent such situations, we give precedence to the balancing fraction and reduce the number of samples appropriately. For the above example, if we sample at most 1000 points, then irrespective of what the preliminary cluster sizes were, it is always possible to satisfy the balancing constraints. In general, sampling a maximum of $\lfloor N(1 - b(1 - \frac{1}{k}))\rfloor$ points always leaves enough points to generate a balanced cluster with balancing fraction $b$. Since $\lfloor N(1-b)\rfloor \leq \lfloor N(1 - b(1 - \frac{1}{k}))\rfloor$, we draw at most $\lfloor N(1-b)\rfloor$ samples in the first step of the reported experiments. Hence, for all the results reported, if $s$ is the attempted number of samples and $s_{\text{true}}$ is the actual number of samples, then $s_{\text{true}} = \lfloor \min(s, N(1-b))\rfloor$. Unless otherwise mentioned, all reported results are for $s = N/2$ so that $s_{\text{true}} = \lfloor \min(N/2, (1-b)N)\rfloor$. For the example above, since $b = 0.9$, we draw $s_{\text{true}} = \lfloor \min(N/2, N/10)\rfloor = \lfloor N/10\rfloor$ samples.

The results on classic3 are shown in Figure 6.1. From the results in Figure 6.1(a), we see that most algorithms perform very similarly under very weak balancing requirements. `KMeans` performs poorly compared to all the other algorithms that are well suited for high-dimensional data. Also, `SPKgpnr` achieves lower NMI values compared to the other `SPKMeans` based algorithms since it uses greedy populate and does not refine the resulting clusters. In Figure 6.1(b), since the balancing fraction is 0.9, the algorithms that satisfy the balancing constraints, viz `SPKpr` and `SPKpnr`, achieve lower NMI than those that do not satisfy the constraints. In particular, `SPKgpr` performs better than its corresponding balanced algorithm `SPKpr`, and `SPKgpnr` performs better than its corresponding `SPKpnr`. This decrease in performance is due to the fact that the balancing algorithms are satisfying the balancing

constraints. In fact, the value of the refinement stage becomes clear as SPKpr, that guarantees balancing, performs better than SPKgpnr, that is unconstrained but does use the refinement phase. Figure 6.1(c) shows the variation in NMI across a range of balancing fractions, with the results of KMeans and SPKMeans shown as points, since their performance does not depend on the balancing fraction. Again, the algorithms respecting balancing constraints perform better over the entire range. Figure 6.1(d) shows the change in standard deviation in cluster sizes as the number of clusters changes. The standard deviation goes down for all the algorithms, although it is marginally more pronounced for the balancing algorithms. Figures 6.1(e) and 6.1(f) show the minimum-to-average ratio of cluster sizes with varying number of clusters. While there is not much difference in Figure 6.1(e) due to the weak balancing requirement, Figure 6.1(f) clearly shows how SPKpr and SPKpnr respect the balancing requirement while the fraction gets small for the other algorithms.

Figure 6.2 shows the results on news-20. This is a typical high-dimensional text clustering problem and the true clusters are balanced. As shown in Figure 6.2(a), the balanced algorithms SPKpr and SPKpnr perform as good as SPKMeans, whereas the unconstrained algorithms SPKgpr and SPKgpnr do not perform as well. Clearly, the balancing constraints resulted in better results. As before, KMeans does not perform as well as the other algorithms. Under a stricter balancing requirement in Figure 6.2(b), as before, SPKgpr performs marginally better than SPKpr, but the latter satisfies the balancing constraints. The same behavior is observed for SPKgpnr and its corresponding SPKpnr. Note that among the two balancing algorithms, SPKpr performs much better than SPKpnr, thereby showing the value of the refinement step. The same is observed for the unbalanced algorithms as well. Figure 6.2(c) shows the variation in NMI across balancing constraints for the right number of clusters. We note that the refined algorithms perform much better, although the constraints do decrease the performance by a little amount.

Interestingly, both KMeans and SPKmeans achieve very low minimum balancing fraction. Figure 6.2(d) shows the standard deviation in cluster sizes. The balancing algorithms achieve the lowest standard deviations, as expected. Figures 6.2(e) and 6.2(f) show the minimum-to-average ratio of cluster sizes. Clearly, the balancing algorithms respect the constraints whereas the ratio gets really low for the other algorithms. For a large number of clusters, almost all the unconstrained algorithms start giving zero-sized clusters.

The overall behavior of the algorithms on small-news-20 (Figure 6.3) and similar-1000 (Figure 6.4) are more or less the same as that described above for news-20 and classic3.

Figure 6.5 shows the results on yahoo. This is a very different dataset from the previous datasets since the natural clusters are highly unbalanced with cluster sizes ranging from 9 to 494. The comparison on most measures of performance look similar to that of other datasets. The major difference is in the minimum-to-average ratio shown in Figures 6.5(e) and 6.5(f). As expected, the balanced algorithms SPKpr and SPKpnr respect the constraints. The other algorithms (except KMeans) start getting zero-sized clusters for quite low values of clusters. Also, as the balancing requirement becomes more strict (as in Figure 6.5(f)), the disparity between the balanced and other algorithms become more pronounced. Surprisingly, even for such an unbalanced data, the balanced algorithms, particularly SPKpr, perform almost as good as the unconstrained algorithms (Figures 6.5(c)).

Figure 6.6 shows the results on nsf-top30. Although this is a rather large dataset with 128,111 points and 30 natural clusters, the performance of the scaled algorithms are quite competitive. In fact, as shown in Figure 6.6(a), at a balancing fraction of 0.3, the refined algorithms SPKpr and SPKgpr often perform better than the base algorithm SPKMeans. At a higher balancing fraction of 0.9, the NMI values for the algorithms SPKpr and SPKpnr guaranteed to give balancing decreases

(Figure 6.6(b)). Even under such a strict balancing requirement, which is violated by the natural clusters themselves, `SPKpr` still performs quite well. At the correct number of clusters, the performance of the algorithms, as shown in Figure 6.6(c), are comparable over a range of balancing fractions. Interestingly, the balancing fractions achieved by `KMeans` and `SPKMeans` are really low as seen in Figure 6.6(c). In fact, `SPKMeans` has a balancing fraction of 0, meaning it generated at least one empty cluster in all 10 runs. Figure 6.6(d) shows the variation in standard deviation in cluster sizes. As expected, the algorithms guaranteed to generate balanced clusters show a lower standard deviation. In Figures 6.6(e) and (f), we present the minimum-to-average ratio of cluster sizes over a range of number of clusters. Other than `SPKpr` and `SPKpnr` which are guaranteed to meet the balancing requirements, all other algorithms generate empty clusters even for moderate number of clusters. Also, the low balancing fractions of `KMeans` and `SPKMeans`, as observed in Figure 6.6(c), is better explained by these figures.

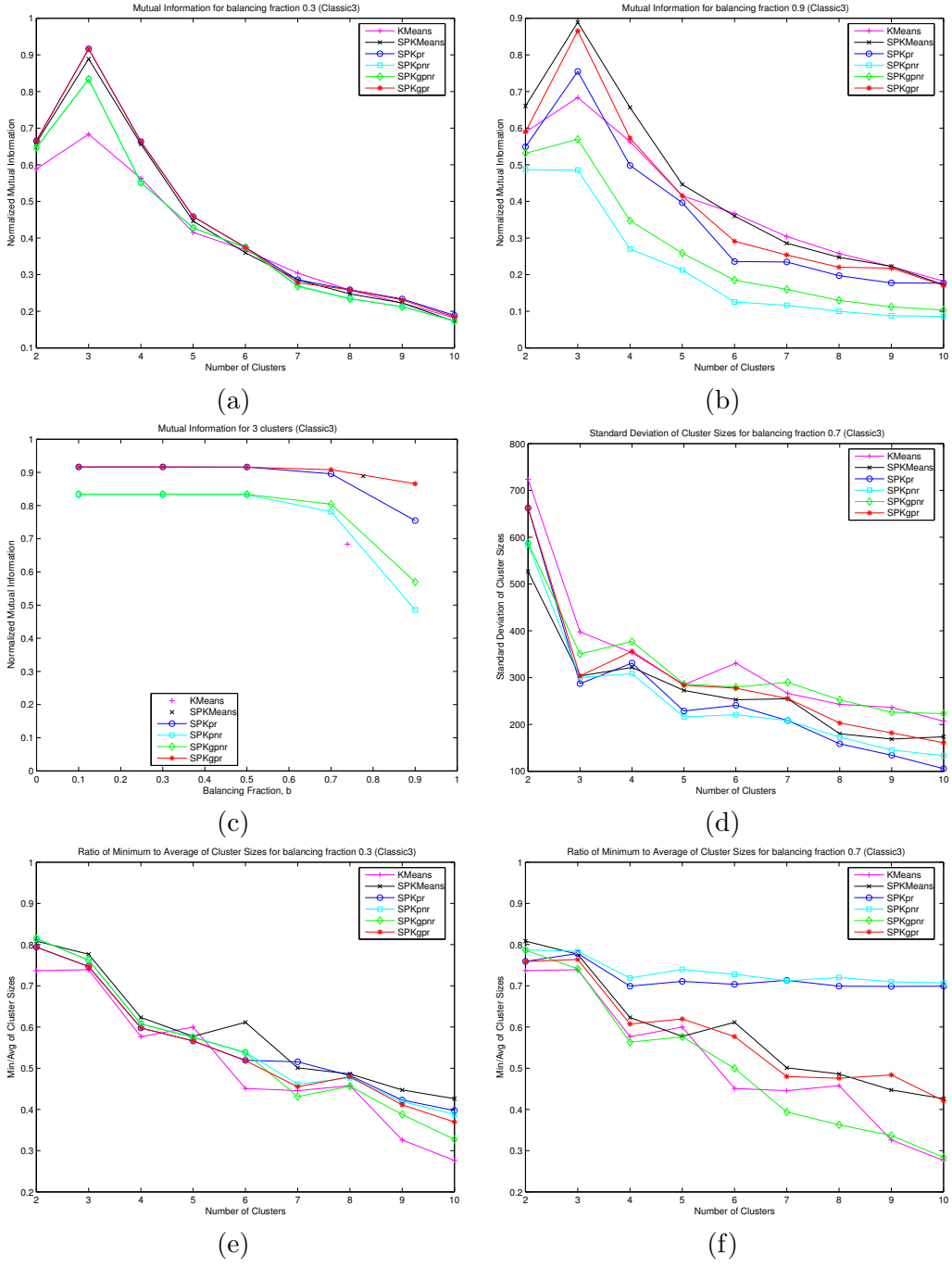


Figure 6.1: Results on classic3: normalized mutual information at balancing fractions (a) 0.3, and (b) 0.9; (c) normalized mutual information for 3 clusters; (d) standard deviation of cluster sizes at balancing fraction 0.7; minimum-to-average ratio of cluster sizes for balancing fractions (e) 0.3, and (f) 0.9.

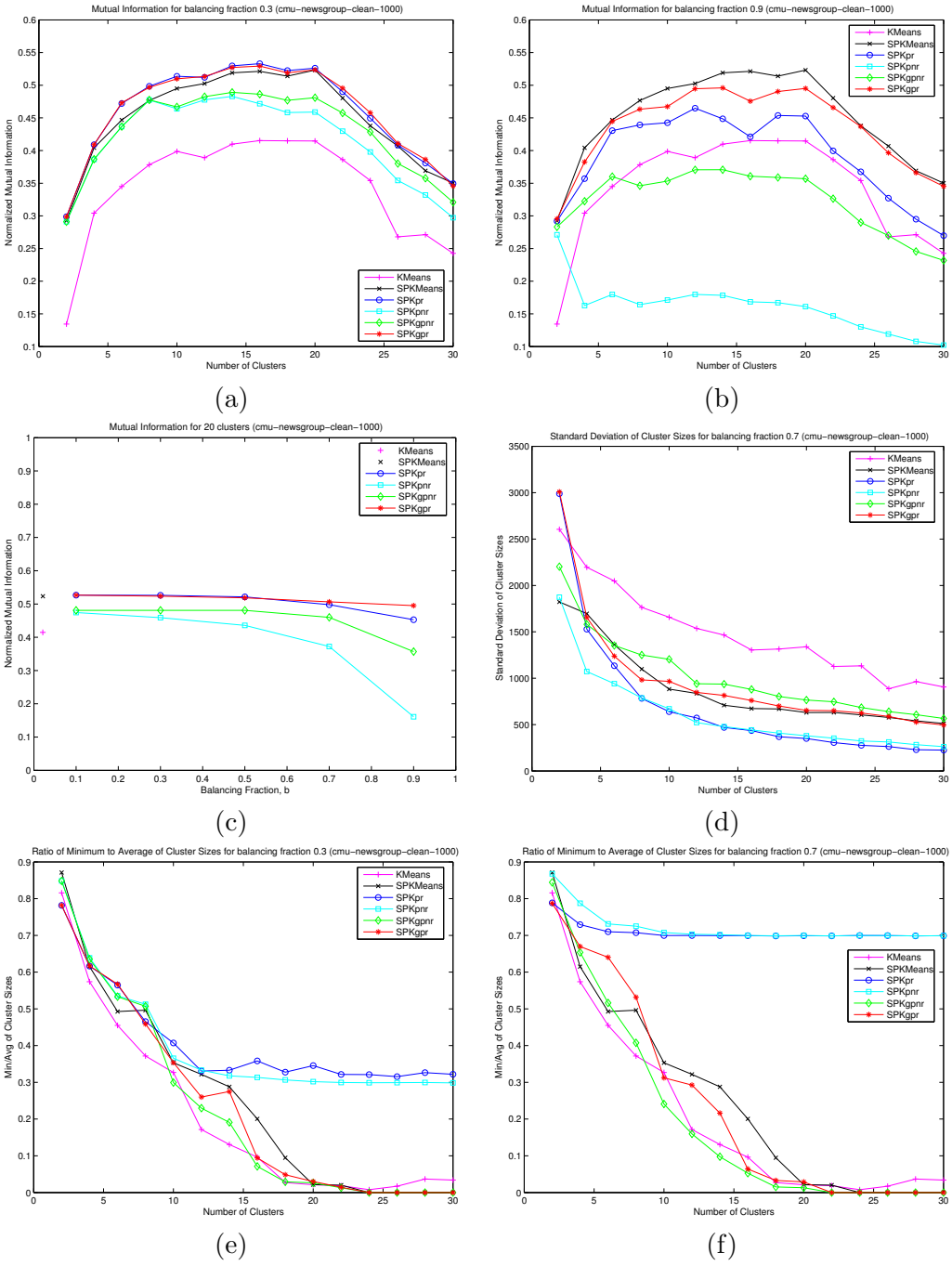


Figure 6.2: Results on news20: normalized mutual information at balancing fractions (a) 0.3, and (b) 0.9; (c) normalized mutual information for 3 clusters; (d) standard deviation of cluster sizes at balancing fraction 0.7; minimum-to-average ratio of cluster sizes for balancing fractions (e) 0.3, and (f) 0.9.

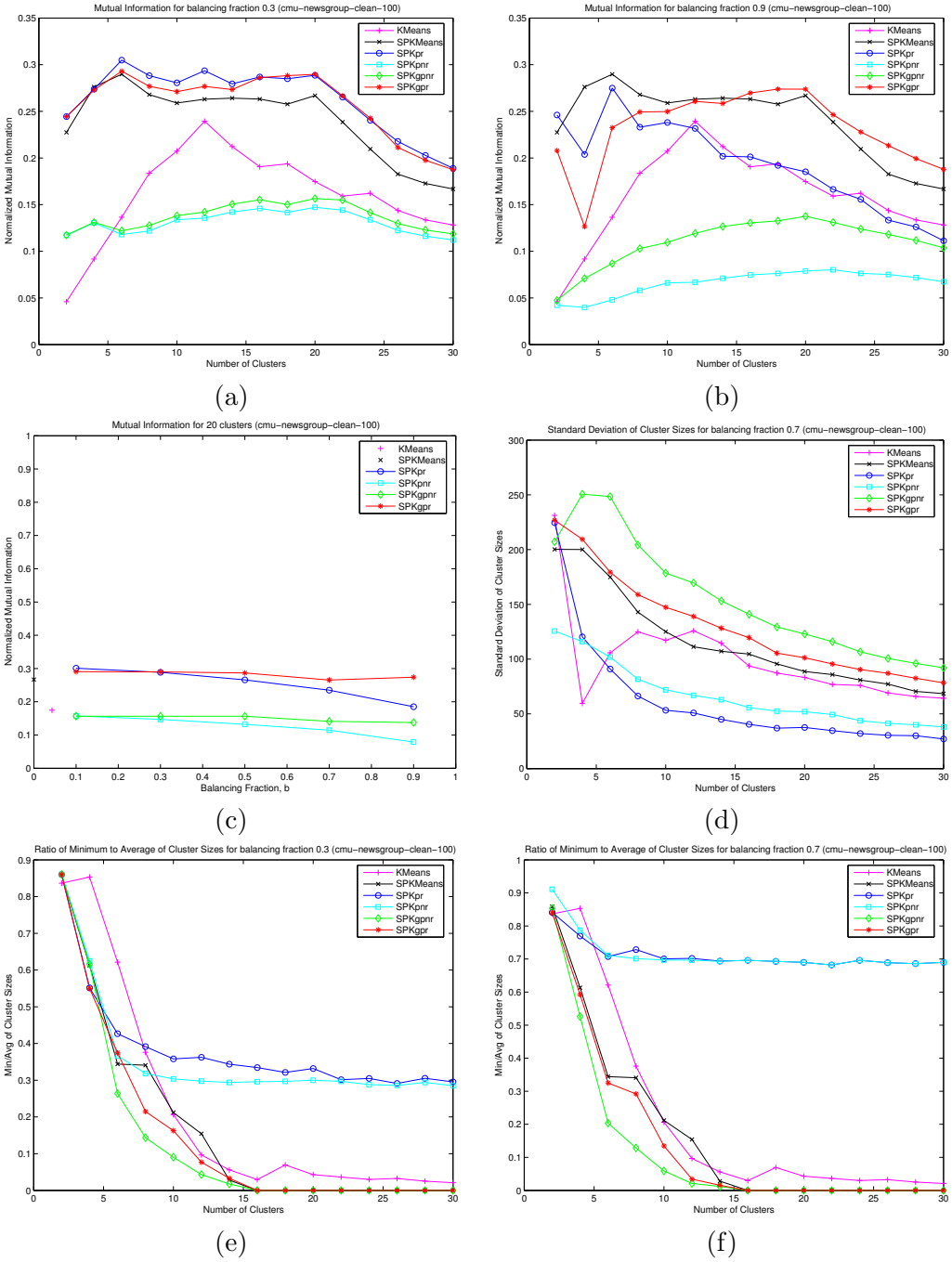


Figure 6.3: Results on small-news20: normalized mutual information at balancing fractions (a) 0.3, and (b) 0.9; (c) normalized mutual information for 3 clusters; (d) standard deviation of cluster sizes at balancing fraction 0.7; minimum-to-average ratio of cluster sizes for balancing fractions (e) 0.3, and (f) 0.9.

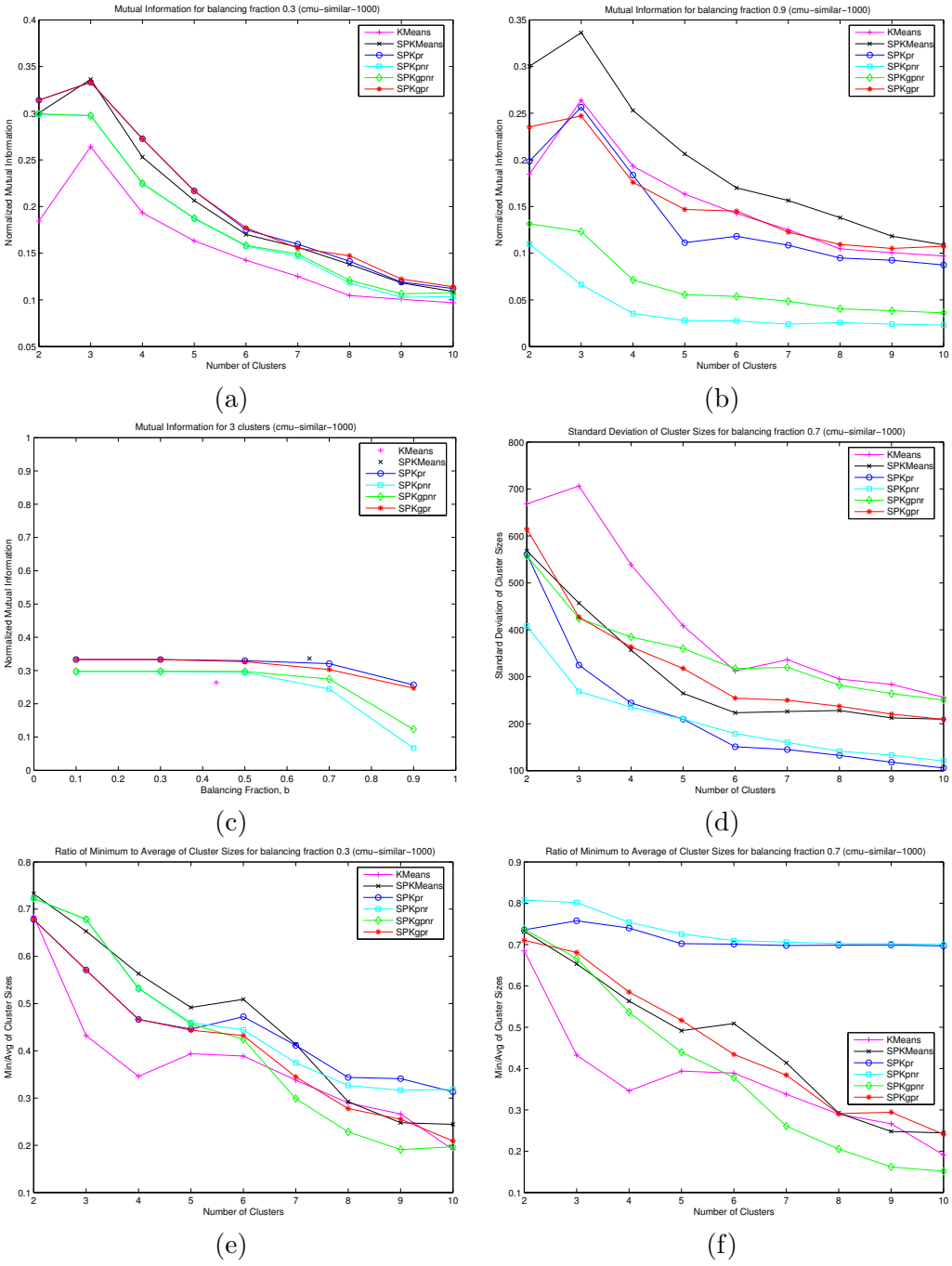


Figure 6.4: Results on similar-1000: normalized mutual information at balancing fractions (a) 0.3, and (b) 0.9; (c) normalized mutual information for 3 clusters; (d) standard deviation of cluster sizes at balancing fraction 0.7; minimum-to-average ratio of cluster sizes for balancing fractions (e) 0.3, and (f) 0.9.

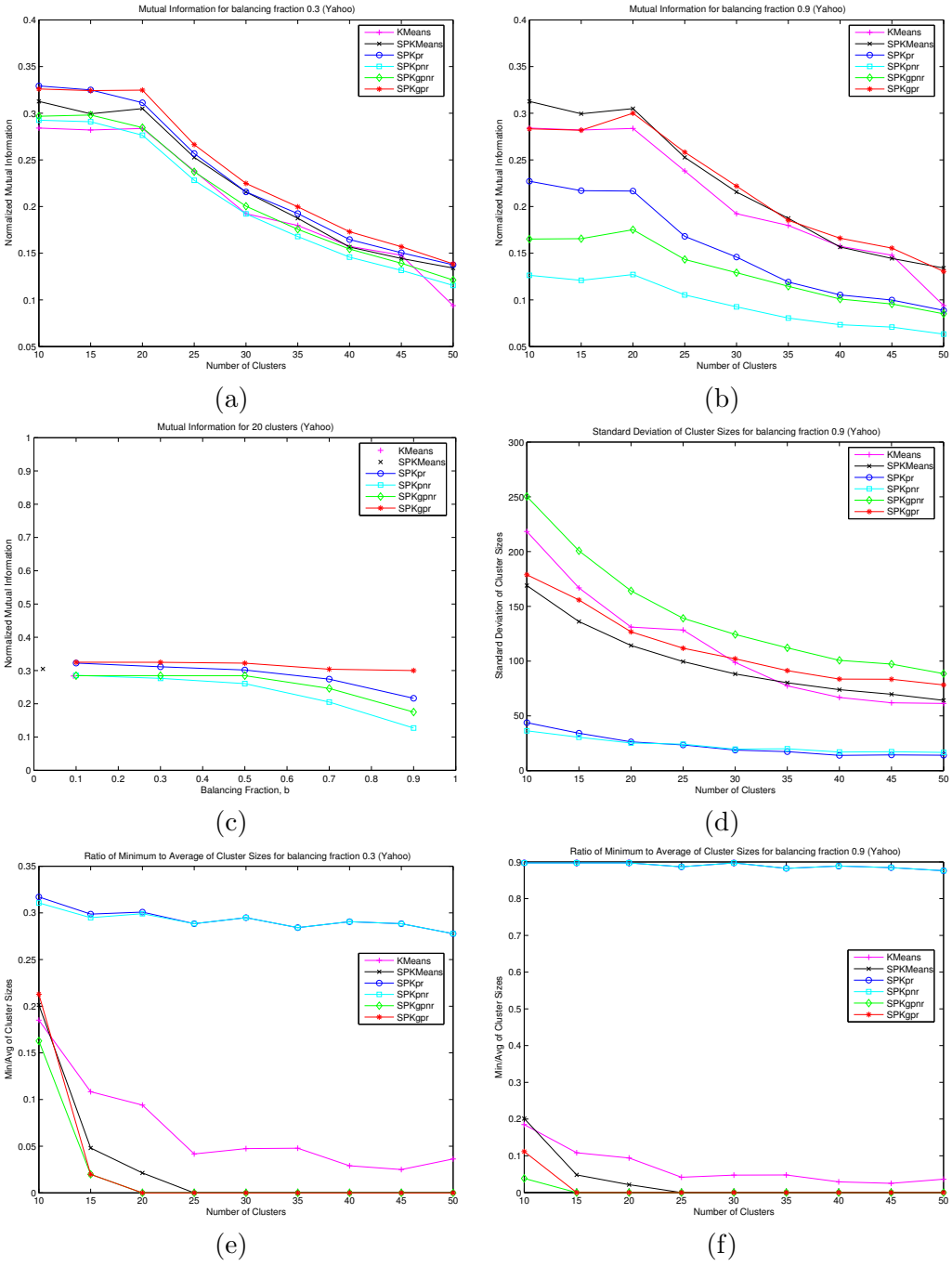


Figure 6.5: Results on yahoo: normalized mutual information at balancing fractions (a) 0.3, and (b) 0.9; (c) normalized mutual information for 3 clusters; (d) standard deviation of cluster sizes at balancing fraction 0.7; minimum-to-average ratio of cluster sizes for balancing fractions (e) 0.3, and (f) 0.9.

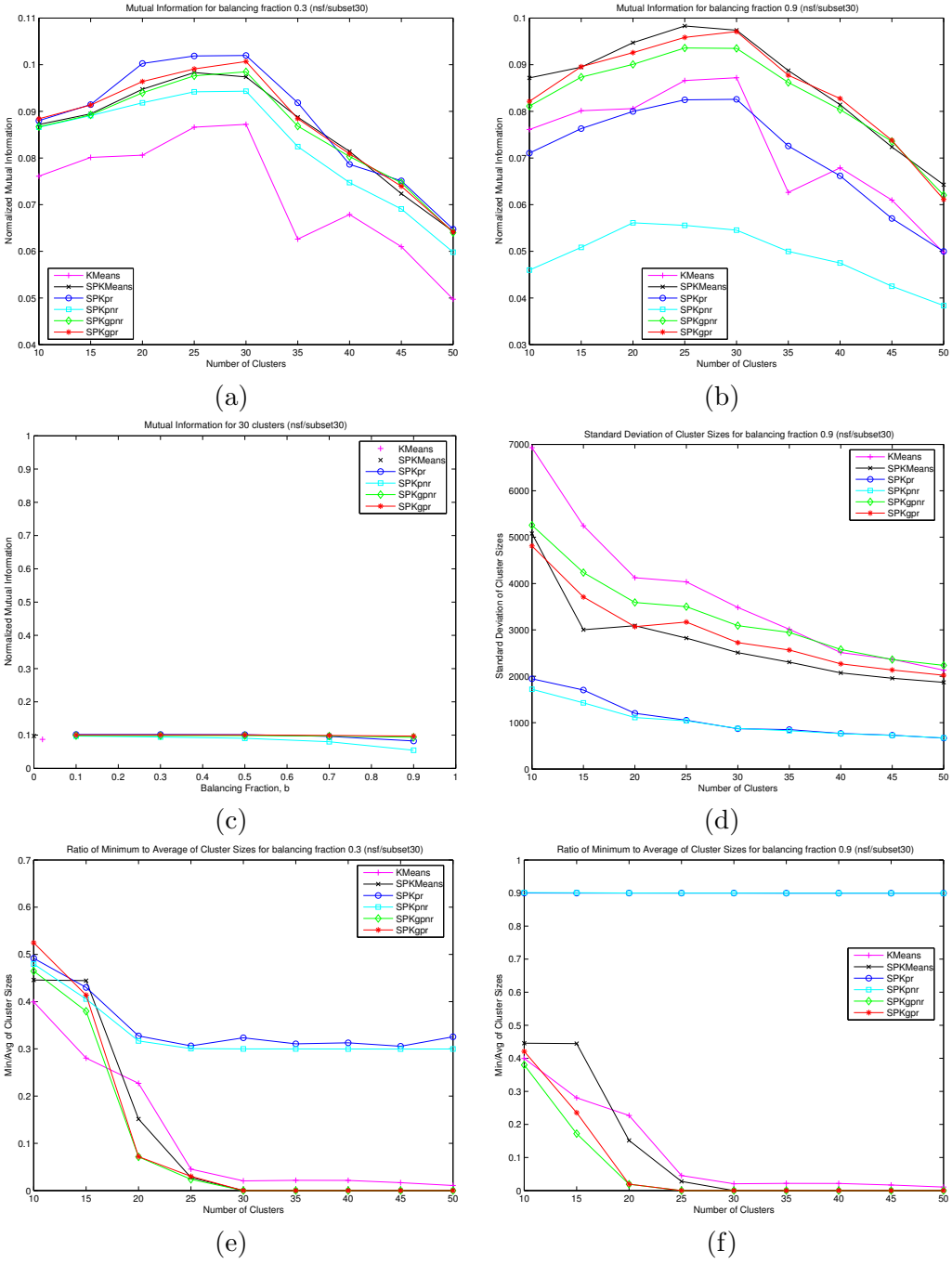


Figure 6.6: Results on nsf-top30: normalized mutual information at balancing fractions (a) 0.3, and (b) 0.9; (c) normalized mutual information for 3 clusters; (d) standard deviation of cluster sizes at balancing fraction 0.7; minimum-to-average ratio of cluster sizes for balancing fractions (e) 0.3, and (f) 0.9.

# Chapter 7

# Evaluation and Model Selection for Clustering

## 7.1 Motivation

In Chapters 2-6, we reviewed as well as presented various types of clustering algorithms suitable for different clustering tasks. Each one optimizes some unsupervised internal quality measure such as minimizing the intra-cluster distances, maximizing the inter-cluster distances, maximizing the log-likelihood of a parametric mixture model, minimizing the cut-value of a pairwise similarity graph, etc. The differences between these internal quality measures make it practically impossible to objectively compare clustering algorithms. Due to the lack of an objective measure, choosing the "right" algorithm for a particular task is confusing.

This chapter introduces a method which eliminates some of these confusions in some settings. Clustering is often used as an intermediate exploratory data analysis step for a prediction problem. Hence, the answer to *what is a good clustering algorithm for my problem?* often depends on how much prediction power the clus-

[0]The work presented in this chapter has earlier appeared as [BL04].

tering step provides, or, more directly, *what is the predictive accuracy of a clustering algorithm?* In other words, since clustering is often used as a method for gaining insight into a dataset, our method here evaluates the degree to which clustering aids the understanding of a dataset *for the purpose of classification*. On natural semi-supervised learning datasets, if the clustering "agrees well" with a set of hidden labels using a small number of clusters, we say that the clustering algorithm is good for prediction. The precise definition of "agrees well" is mediated by the PAC-MDL bound [BL03] on the accuracy of a test set. Our proposed criterion has several natural properties:

1. It applies to every clustering algorithm.
2. It is inherently normalized to the interval $[0, 1]$.
3. All possible values are exercised on natural datasets.
4. The metric can flexibly incorporate prior information by proper design of a *description language*.
5. It can be used for model selection.
6. It is directly related to a concrete goal (good prediction).

It is important to note that this is not the only possible objective criterion for evaluation of clustering algorithms. Clustering algorithms may be used for other purposes, and when so used, other measures may be more appropriate. Our results here are only directly applicable when the goal in clustering is related to prediction.

The rest of the chapter is organized as follows. First, we explain the PAC-MDL bound [BL03] in section 7.2. In section 7.3, we discuss how the PAC-MDL bound can be applied to a clustering setting for performance evaluation and model selection. We present experimental results on benchmark text datasets in section 7.4

to demonstrate the proposed approach. We end with a discussion on the proposed criterion in section 7.5.

## 7.2 The PAC-MDL Bound

The PAC-MDL bound is the core mechanism used to trade off between a sufficiently rich representation to capture the data and over-fitting on the data. Clustering algorithms with a small bound must have a small number of clusters which agree well with a set of (hidden) labels.

Consider the following learning setting: Let $D$ be any distribution over $(X, Y)$ where $X$ denotes the input and $Y$ denotes the label. We assume $Y$ can take one of $\ell > 1$ possible values, i.e., $Y \in \{1, \cdots, \ell\}$. Consider a train set $S = (X^m, Y^m)$ and a test set $S' = (X^n, Y^n)$, where $(X^i, Y^i)$ denotes the set of $i$ independently drawn samples from the (unknown) joint distribution $D$ over $(X, Y)$. The PAC-MDL bound applies to a transductive learning algorithm, $T : (X \times Y)^m \times X^n \mapsto \sigma$, where $\sigma : X^{m+n} \mapsto \hat{Y}^{m+n}$ is a transductive classifier which produces a simultaneous labeling $\hat{Y}^{m+n}$ for all of the data points $X^{m+n}$. Given the description complexity of the transductive classifier measured by its bit description length $|\sigma|$, and the prediction error count on the train set, $\hat{\sigma}_S = |\{Y^m \neq \hat{Y}^m\}|$, the bound limits the error count on the test set $\hat{\sigma}_{S'} = |\{Y^n \neq \hat{Y}^n\}|$.

The precise bound is defined in terms of a cumulative hypergeometric distribution. To understand this distribution, imagine a bucket with $m$ red balls and $n$ blue balls, from which $(a + b)$ balls are drawn without replacement. Now, define Bucket$(m, n, a, b)$ to be the probability that at least $b$ blue balls are drawn. That is,

$$\text{Bucket}(m, n, a, b) \quad = \quad \sum_{t=b}^{a+b} \frac{\binom{n}{t}\binom{m}{a+b-t}}{\binom{n+m}{a+b}}.$$

The bound is actually defined in terms of a "worst case" over the value of $b$ defined

according to:

$$\text{bmax}(m, n, a, \delta) \quad = \quad \max\{b : \text{Bucket}(m, n, a, b) \geq \delta\}$$

Thus, for any $b > \text{bmax}(m, n, a, \delta)$, if $(a + b)$ balls are drawn out of the $(m + n)$ balls, the probability of getting at least $b$ blue balls is less than $\delta$. In the PAC-MDL bound, $a$ plays the role of $\hat{\sigma}_S$, the number of errors on the train set, and $b$ plays the role of $\hat{\sigma}_{S'}$, the number of errors on the test set, which is exactly the number we wish to bound.

**Theorem 8 (PAC-MDL bound [BL03])** *For any distribution $D$, for any label description language $L = \{\sigma\}$, with probability at least $(1 - \delta)$ over the draw of the train and test sets $S, S' \sim D^{m+n}$:* $\quad \forall \sigma, \quad \hat{\sigma}_{S'} \leq \text{bmax}(m, n, \hat{\sigma}_S, 2^{-|\sigma|}\delta)\}$

Intuitively, the theorem says that if a transductive classifier with a short description length achieves few errors on the train set, then the number of errors on the test set is small with high probability. For details on this theorem, its proof, and its connection to other PAC bounds, see [BL03]. For now, it is important to note that the applicability of this theorem is only limited by the assumption that the train and test sets are each drawn independently from the distribution $D$, and that $L = \{\sigma\}$ is a "valid" description language.

## 7.3 Application to Clustering

In this section, we discuss the application of the PAC-MDL bound to clustering. Two important issues are relevant for clustering bounds:

A. How does a clustering algorithm produce a prediction?

B. What is a "valid" description language for transductive classifiers?

For A, recall that the PAC-MDL bound is applicable to a transductive classifier. It turns out that *any* clustering can be converted into a transductive classifier. Given the entire train set $(X^m, Y^m)$, and $X^n$ from the test set, consider $X^{m+n}$ as the input to the clustering algorithm. Say the clustering algorithm partitions $X^{m+n}$ into $k$ subsets. To find the labels $\hat{Y}^{m+n}$ on all points, first compute the most common label in each cluster using $Y^m$. Then,

i. if the most common is of class $i$, $i \in 1, \cdots, \ell$, label *all* points in that cluster as class $i$;

ii. if there is a tie between two or more class labels, pick one of them uniformly at random and label *all* points in that cluster with that label;

iii. if there are no labeled points in a cluster, choose a label $i \in \{1, \cdots, \ell\}$ uniformly at random and label *all* points in that cluster with that label.

After the assignment of the new labels, all points in each cluster have the same label.

For issue B above, note that the description $\sigma$ of a classifier can be considered a binary code that contains all the relevant information for generating the labels $\hat{Y}^{m+n}$. Hence, formally speaking, there is a Turing machine that takes $\sigma$ as an input, produces the labels $\hat{Y}^{m+n}$ as output and halts. Therefore, the description language $L = \{\sigma\}$ must be a prefix-free code (by the halting property) and hence satisfy Kraft's inequality (see [CT91], Theorem 5.2.1, Lemma 7.3.1, for details), i.e., $\sum_{\sigma \in L} 2^{-|\sigma|} \leq 1$. This is the *only* condition a label description language $L = \{\sigma\}$ must satisfy for the proposed bound to hold.

### 7.3.1 PAC-MDL Bound for Clustering

Consider the problem of clustering a dataset $X^{m+n}$, where $m$ is the train-set size and $n$ is the test-set size. For any fixed[1] clustering algorithm, we can construct a

[1]We mean fixed initialization and fixed cluster number here.

description language $L = \{\sigma\}$ by letting each description $\sigma$ have a constant label on each cluster, as discussed earlier. Given the clustering, this description is sufficient to generate the new labels $\hat{Y}^{m+n}$ over the entire data. Now, if $c$ is the number of clusters and $\ell$ is the number of labels, then the set of descriptions, i.e., the language $L = \{\sigma\}$, has size at most $\ell^c$ which can be indexed using only $|\sigma| = c \log \ell$ bits ("fractional bits" are ok here). Since $|L| \leq \ell^c$, it is straightforward to see that Kraft's inequality is indeed satisfied, and hence $L$ is a valid description language. On a more general note, if one assigns a probability measure $p(\sigma)$ to all $\sigma \in L$, since

$$\sum_{\sigma \in L} p(\sigma) = 1 \quad \Rightarrow \quad \sum_{\sigma \in L} 2^{-\log\left(\frac{1}{p(\sigma)}\right)} = 1 \ ,$$

$|\sigma| = \log\left(\frac{1}{p(\sigma)}\right)$ always gives a valid bit description complexity for $\sigma$. Here, since $|L| = \ell^c$, we have essentially assigned a uniform probability of $p(\sigma) = \frac{1}{\ell^c}, \ \forall \sigma \in L$.

We call the above description language Simple. Using a direct application of the PAC-MDL bound, we get: With probability at least $(1 - \delta)$ over the draw of the train and test sets $S, S' \sim D^{m+n}$, the test-set error is bounded by:

$$\hat{\sigma}_{S'} \leq \text{bmax}\left(m, n, \hat{\sigma}_S, \frac{\delta}{l^c}\right) \tag{7.1}$$

In practice, clustering algorithms are highly dependent upon random initializations, so the algorithm is run multiple times with the best[2] performing run chosen. Note that the description length of this scheme is higher since the description language must specify which random initialization to use. If the best initialization is chosen from $r$ random initializations, the description length is $|\sigma| = c \log \ell + \log r$. We call this language Init. Applying the PAC-MDL bound for Init, we get with

[2] "Best" might be defined with respect to some algorithm-specific metric or with respect to bound performance, depending on what you want to evaluate.

probability at least $(1-\delta)$ over the draw of the train and test sets:

$$\hat{\sigma}_{S'} \leq \text{bmax}\left(m, n, \hat{\sigma}_S, \frac{\delta}{rl^c}\right) \tag{7.2}$$

### 7.3.2 The Right Number of Clusters

Most clustering algorithms need the number of clusters as an input to the algorithm. The PAC-MDL bound provides a natural mechanism for cluster number selection when some label information is available. Consider running a clustering algorithm on a dataset over a range of cluster numbers and picking the cluster number with the tightest bound on the test-set error. This process increases the size of our set of descriptions again. We use a description language for the cluster number $c$ requiring $\log c(c+1)$ bits. This is "legal" because it does not violate the Kraft inequality since: $\sum_{c=1}^{\infty} \frac{1}{c(c+1)} = 1$. One slight optimization is possible here: the nature of our metric disallows the $c = 1$ case, implying that we can substitute $c \to c-1$. With this description language, the length of our description is $|\sigma| = c \log \ell + \log r + \log(c(c-1))$ bits. We call the language Cluster. Applying the PAC-MDL bound for Cluster, we get that with probability at least $(1-\delta)$ over the draw of the train and test sets, the optimal cluster number $c^*$ achieves a test-set error of:

$$\hat{\sigma}_{S'} \leq \text{bmax}\left(m, n, \hat{\sigma}_S, \frac{\delta}{rl^{c^*}c^*(c^*-1)}\right) \tag{7.3}$$

### 7.3.3 The Right Algorithm

In practice, for a given dataset, it is not clear which clustering algorithm is appropriate for use. Typically an algorithm is chosen using domain knowledge, and there is normally no objective way to verify whether the choice made was good or bad. To cope with this, we can extend our description language to specify one of multiple algorithms. More precisely, if we are choosing the best among $s$ clustering algorithms,

an extra $\log s$ bits are required to send the index of the clustering algorithm that performs the best. Hence, $|\sigma| = c^* \log \ell + \log r + \log((c^*-1)c^*) + \log s$. Being the best over all algorithms, we call this language Algo. Applying the PAC-MDL bound, we see that with probability at least $(1-\delta)$ over the draw of the train and test sets, the best algorithm with the optimal cluster number and optimal initialization achieves a test-set error of:

$$\hat{\sigma}_{S'} \leq b_{\max}\left(m, n, \hat{\sigma}_S, \frac{\delta}{r\ell^{c^*}c^*(c^*-1)s}\right) \ . \tag{7.4}$$

## 7.4 Experimental Results

We present an empirical study of the proposed evaluation technique on the problem of text clustering for several benchmark datasets using various algorithms.

### 7.4.1 Datasets

The datasets that we used for empirical validation and comparison of our algorithms were carefully selected to represent some typical clustering problems: (a) classic3 is a well known collection of documents that contains 3893 documents, among which 1400 Cranfield documents are from aeronautical system papers, 1033 Medline documents are from medical journals, and 1460 Cisi documents are from information retrieval papers; (b) classic2 is a subset of 2860 documents from the classic3 collection formed with the 1400 Cranfield documents and the 1460 Cisi documents; (c) cmu-newsgroup-clean-1000, or, the CMU 20 newsgroups dataset is a widely used text analysis dataset that is a collection of approximately 20,000 messages from 20 different USENET newsgroups, with approximately 1000 messages per group; (d) cmu-newsgroup-clean-100 was formed by sampling 100 messages per group from the full 20 newsgroup dataset; (e) cmu-different-1000 is a subset of the original 20 newsgroups dataset consisting of 3 groups on very differ-

ent topics: *alt.atheism*, *rec.sport.baseball*, *sci.space*; (f) cmu-different-100 is a subset of (e) formed by sampling 100 documents per topic; (g) cmu-similar-1000 is a subset of the original 20 newsgroups dataset consisting of 3 groups on similar topics: *talk.politics.guns*, *talk.politics.mideast*, *talk.politics.misc*; (h) cmu-similar-100 is a subset of (g) formed by taking 100 documents per topic; (i) cmu-same-1000 is a subset of the original 20 newsgroups dataset consisting of 3 groups on the same topic viz computers, with different subtopics : *comp.graphics*, *comp.os.ms-windows*, *comp.windows.x*; (j) cmu-same-100 is a subset of (i) formed by sampling 100 documents per topic; and, (k) yahoo, or, the Yahoo News (K-series) dataset that has 2340 Yahoo news articles from 20 different categories.

### 7.4.2 Algorithms

We experiment with 6 algorithms that have been applied to text datasets with varying degrees of success. Since the motivation behind the experiments is to establish the efficacy of the proposed criterion in evaluation, comparison and model selection for clustering, we have not tried to be exhaustive in the list of algorithms that have been used. However, we have chosen algorithms that represent the state-of-the-art and have been applied to text clustering in the literature. The algorithms we consider are: `SPKMeans` [DM01], better known as spherical kmeans, that employs the widely used cosine similarity; `FSKMeans` [BG04], a frequency sensitive version of spherical kmeans; `Hard-moVMF` [BDGS03], a generative model based clustering that uses a mixture of von Mises-Fisher (vMF) distributions to model the data; `Soft-moVMF` [BDGS03], that also uses a mixture of von Mises-Fisher distributions to model the data with soft-assignments that are finally converted to hard assignments by the standard method assigning a data point to the highest probability cluster; `KMeans` [JD88], the standard kmeans clustering algorithm; and `KLKMeans` [DMK03], better known as information theoretic clustering, that uses KL-

divergence between $L_1$ normalized document vectors instead of squared Euclidean distance in the kmeans framework.

### 7.4.3 Methodology

Before performing any experiments, an independent train-test split needs to be made. All experiments reported in this chapter were performed on 5 different train-test splits: 10-90, 30-70, 50-50, 70-30, 90-10. On each train-test split, we performed 4 sets of experiments for each dataset corresponding to the 4 bounds discussed in section 3:

(a) Experiments for a particular algorithm, with a fixed cluster number, with a particular initialization. The test-set error-rate bound is computed using (7.1).

(b) The best results for a particular algorithm with a fixed cluster number, where the best is computed over all the $r$ possible initializations. The test-set error-rate bound is computed using (7.2).

(c) The best results for a particular algorithm, where the best is computed over all the $r$ possible initializations and all the $c_{\max}$ possible cluster numbers. The test-set error-rate bound is computed using (7.3).

(d) The best performance for a given dataset, where the best is computed over all algorithms over all possible cluster sizes and all possible initializations. The bound on the test-set error-rate is computed using (7.4).

### 7.4.4 Results

Now we are ready to present results on the various datasets. We start with the simplest language and using (7.1) present representative results comparing individual runs of a particular algorithm for a fixed cluster number with different random

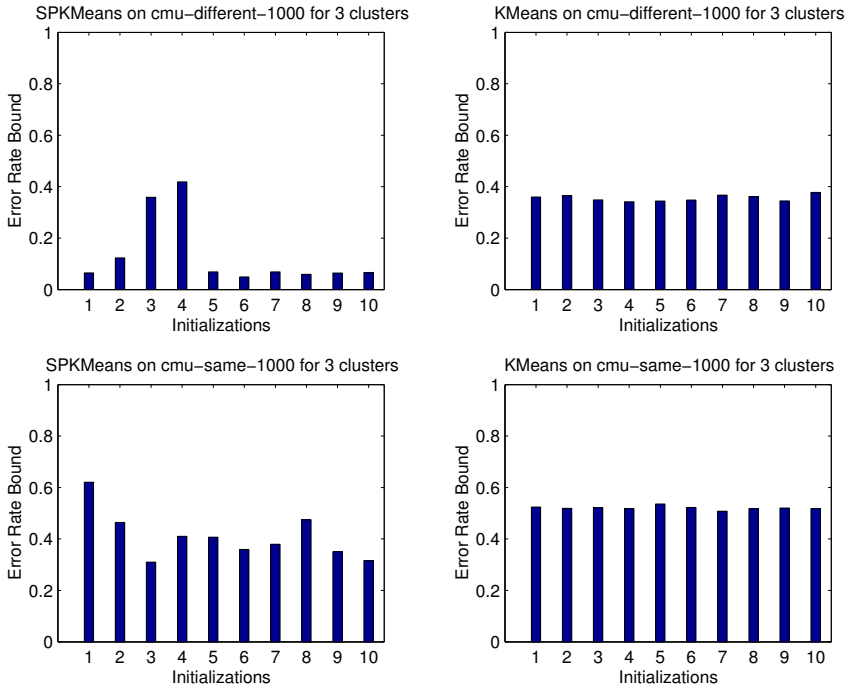


Figure 7.1: Test set error rate bounds for `KMeans` and `SPKMeans` on cmu-different-1000 and cmu-same-1000: 10 runs with different initializations for 3 clusters

initializations (Figure 7.1). Taking the best over 10 initializations for each cluster number, we then compare performance of a particular algorithm over a range of cluster numbers (Figure 7.2) using (7.2). Next, by taking the best over the entire cluster number range considered, we compare the performance of various algorithms (Figure 7.3) using (7.3). Finally, by taking the best over the best of all the algorithms considered, using (7.4), we present the best results on particular datasets for various train-test splits (Figure 7.4-7.5). Unless otherwise stated, all results are on a 50-50 train-test split.

In Figure 7.1, we present test-set error-rate bounds for `KMeans` and `SPKMeans` on cmu-different-1000 and cmu-same-1000 for 10 runs with different initializations for 3 clusters. All bounds are computed using (7.1). cmu-different-1000 is a relatively easy dataset in that its labels are reasonably separated being samples from 3

quite different newsgroups. As a result, both algorithms achieve low bounds on the error-rate. SPKMeans performs particularly well since it was designed to be a text clustering algorithm [DM01]. Over the 10 runs, KMeans achieves a lowest bound of 34.13 % with probability 0.9 in run 4, and SPKMeans achieves a lowest bound of 4.93 % with probability 0.9 in run 6. The best constant classifier has error-rate 66.67 %. cmu-same-1000 is a relatively difficult dataset since the true labels have significant overlaps. Again, SPKMeans achieves lower bounds than KMeans in most runs, although the bounds are in general higher than those for cmu-different-1000. Over the 10 runs, KMeans achieves a lowest bound of 50.8 % with probability 0.9 in run 7, and SPKMeans achieves a lowest bound of 31 % with probability 0.9 in run 3, the best constant classifier has error-rate 66.67 %.

Next, we consider the best performance over all the 10 iterations for each cluster number and compare them over a range of cluster numbers. The bound calculation is done using (7.2). In Figure 7.2, we present test-set error-rate bounds for KMeans and SPKMeans on cmu-different-1000 and cmu-same-1000 over a range of cluster numbers (2 to 20). We observe that SPKMeans achieves a lower bound than KMeans for most cluster numbers for reasons described above. Over the entire range, SPKMeans achieves a lowest bound of 5.46 % with a probability of 0.9 for 3 clusters. This is marginally higher than the lowest bound of 4.93 % in Figure 7.1, since the bound calculation uses (7.2) after incorporating the extra $\log 10$ bits required to index the best over the 10 runs with different random initializations. This is the extra cost for not knowing upfront which random initialization is going to perform the best. This demonstrates the trade-off between improvement in prediction accuracy and considering more runs with different random initializations. On the other hand, KMeans achieves a lowest bound of 17.2 % with a probability of 0.9 for 13 clusters. For cmu-same-1000, over the entire range, SPKMeans achieves a lowest bound of 32.2 % with probability 0.9 for 3 clusters, which is marginally higher than the lowest of

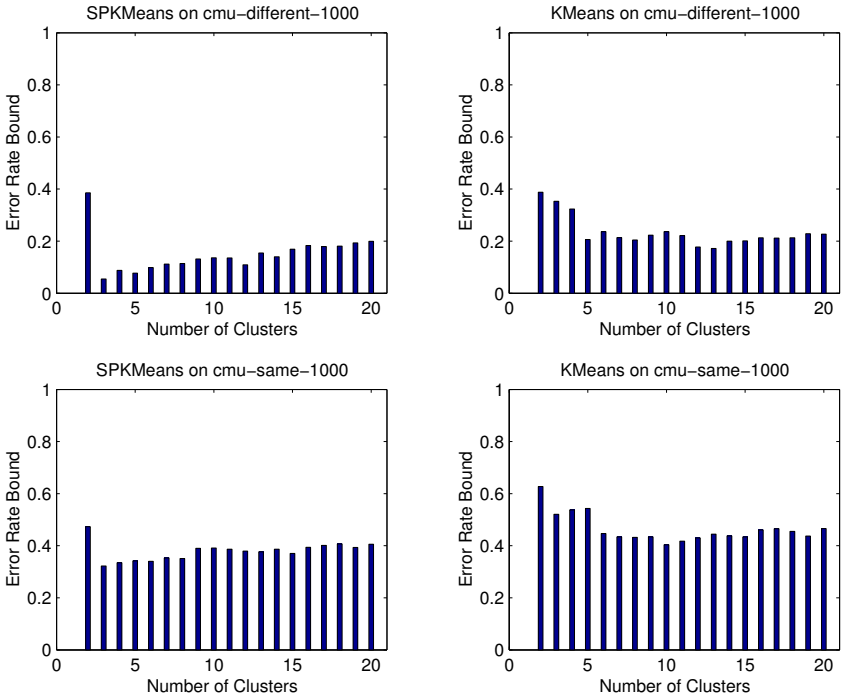


Figure 7.2: Test-set error-rate bounds for `KMeans` and `SPKMeans` on cmu-different-1000 and cmu-same-1000: Best over 10 runs with different initializations over a cluster number range of (2 to 20).

31 % in Figure 7.1 due to extra description complexity of $\log 10$ bits in indexing the best. `KMeans` achieves a lowest bound of 40.3 % with probability 0.9 for 10 clusters.

From the results in Figure 7.2, we make an interesting observation. Note that for `SPKMeans`, the *optimal number of clusters*, as dictated by the lowest bound over the entire range of cluster numbers considered, is 3 for both the datasets. Interestingly, the number of true labels in both the datasets is 3. This demonstrates how the proposed criterion can be used for *model-selection*, the "right" number of clusters in this particular case, for a given algorithm and a dataset.

Next, we compare the best performance of each of the algorithms, with best taken over all cluster numbers and initializations, on various datasets. This com-

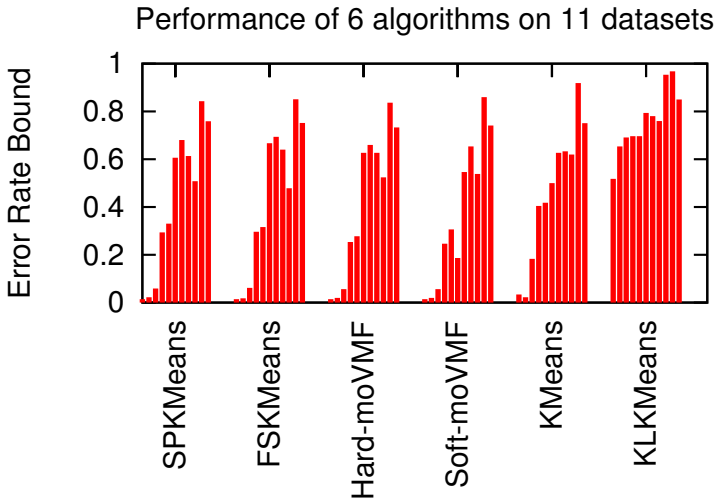


Figure 7.3: Test-set error-rate bounds for each algorithm on all the 11 datasets: classic2, calssic3, cmu-different-1000, cmu-similar-1000, cmu-same-1000, cmu-different-100, cmu-similar-100, cmu-same-100, cmu-newsgroup-clean-1000, cmu-newgroup-clean-100, yahoo: Best over all number of clusters and initializations.

parison is of great practical interest since this determines the appropriateness of an algorithm for a given dataset. Since the best performance of each algorithm over cluster numbers and initializations is considered, (7.3) is used to compute the bounds in Figure 7.3. In Figure 7.3, we compare the test-set error-rate bounds for all the 6 algorithms on all the 11 datasets under consideration. For datasets such as cmu-different-100, cmu-similar-100, cmu-same-100, the low number of samples in high-dimensions make the clustering problem hard for most algorithms. Among the algorithms considered, `Soft-moVMF` performs quite well, e.g., it achieves the lowest test-set error-rate bound of 18.6 % with a probability of 0.9 on cmu-different-100. On the other hand, datasets such as cmu-different-1000 has more samples from the same distribution that makes the clustering problem reasonably simple for quite a few algorithms. We note that 4 algorithms have comparative performances, with `Soft-moVMF` achieving the lowest bound of 5.67 % with a probability of 0.9. It is interesting to note that `SPKMeans` achieves a bound of 5.87 % which is marginally

higher than the bound of 5.46 % we observed in Figure 7.2. The marginal increase is due to the extra description length of $\log((3-1)3)$ bits used to describe the fact that the optimal cluster number is 3. As for the relative performance of the algorithms, as expected, there is no clear winner across all datasets although `Soft-moVMF` appears to win quite often.

We now present best bounds on the test-set error-rate by taking the best over all algorithms, cluster numbers and initializations. We use (7.4) to compute the bound. Results over 5 different train-test splits on all the datasets considered are presented in Figures 7.4-7.5.

classic2 is a relatively simple dataset with only 2 reasonably separate classes. As we see in Figure 7.4, for all train-test splits, the bound on the test-set error-rate is very low. For the 50-50 train-test split, with probability 0.9 we get a PAC bound of 1.68% on the error-rate on the test-set. (The best constant classifier has error-rate of 48.95%.) This is a remarkably low error-rate bound by PAC standards[3]. classic3 is also a relatively simple dataset with 3 classes. As shown in Figure 7.4, for the 50-50 train-test split, with probability 0.9 we get a bound of 2% on the error-rate on the test-set. (The best constant classifier has error-rate 62.50%.)

Although cmu-different-100 consists of samples from 3 relatively different classes, the small number of samples and high dimensionality make the problem difficult. As shown in Figure 7.4, with a probability of 0.9, we get a bound of 20.6% on the test-set error-rate for the 50-50 train-test split. (The best constant classifier has error-rate 66.67%.) In cmu-different-1000, the extra samples make finding structure in the data easier — a bound of 6 % is obtained.

Due to a large overlap between the underlying labels, both cmu-same-100 and cmu-same-1000 are difficult datasets to get good predictions on by just clustering. As shown in Figure 7.5, with a probability of 0.9, we achieve error-rate bounds of

[3]Note that increasing the train-set fraction need not increase prediction accuracy since the clustering algorithms never actually look at the labels.

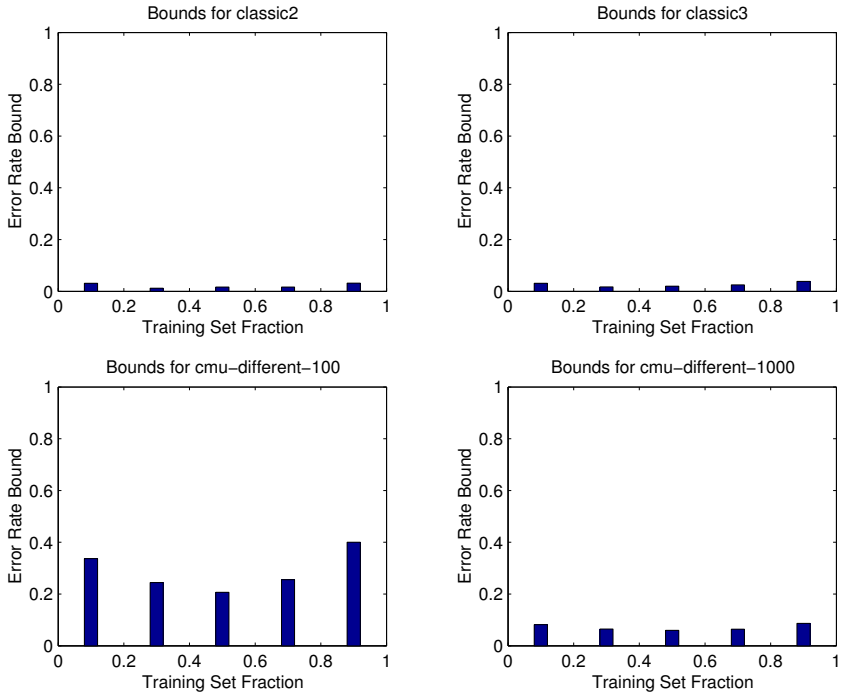


Figure 7.4: Test set error rate bounds for classic2, classic3, cmu-different-100, cmu-different-1000: Best over all algorithms, cluster numbers, initializations

64% and 28.4% respectively, while the best constant classifier has error-rate 66.67%.

cmu-newsgroup-clean-100 is a difficult dataset since there are 20 underlying classes with significant overlaps and small number of samples per class. As Figure 7.5 shows, the lowest bound on the test-set error-rate is 84 % with probability 0.9 on a 50-50 train test split, whereas the best constant classifier has an error-rate of 95 %. For cmu-newsgroup-clean-1000, with increased number of samples from the same problem, a lowest bound of 47.94 % is achieved with probability 0.9.

yahoo is a dataset with 20 underlying classes and 2340 examples. The class portions are highly skewed ranging from as low as 0.0038 to as high as 0.2111. Naturally, unsupervised prediction is nontrivial. The best performance (not shown) achieves a bound of 73.84% with probability 0.9 on the 50-50 train-test split. (The best constant classifier has error-rate 78.89%).

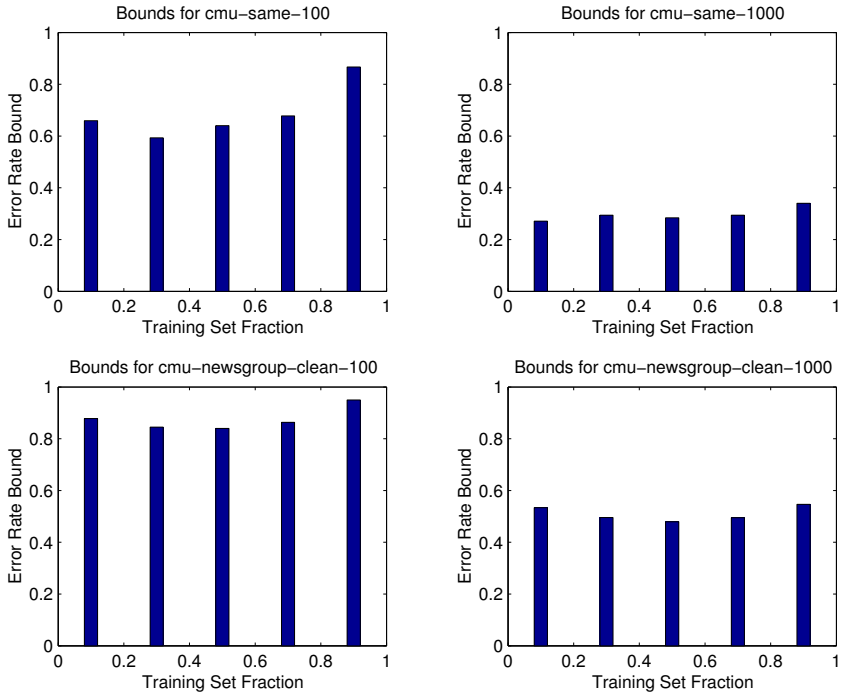


Figure 7.5: Test set error rate bounds for cmu-same-100, cmu-same-1000, cmu-newsgroup-clean-100 and cmu-newsgroup-clean-1000: Best over all algorithms, cluster numbers, initializations

At this point, we make two observations: (i) on all the datasets considered, the test-set error-rate bound is *always* better than the best constant classifier; and, more interestingly, (ii) for most datasets, the bound on the test-set error-rate for the best clustering algorithm is *comparable*, perhaps even *better* than many supervised learning bounds. Clustering appears to capture the structure of labeling in natural datasets.

Finally, we present a comparison between the bounds obtained from the 4 language families considered: Simple, Init, Cluster and Algo, corresponding to (7.1)-(7.4). It is difficult to directly compare languages because each optimizes over a different set of possibilities. For example, it would be unfair to compare the best bound (across all possibilities) for Init to the best bound (across all possibilities) for

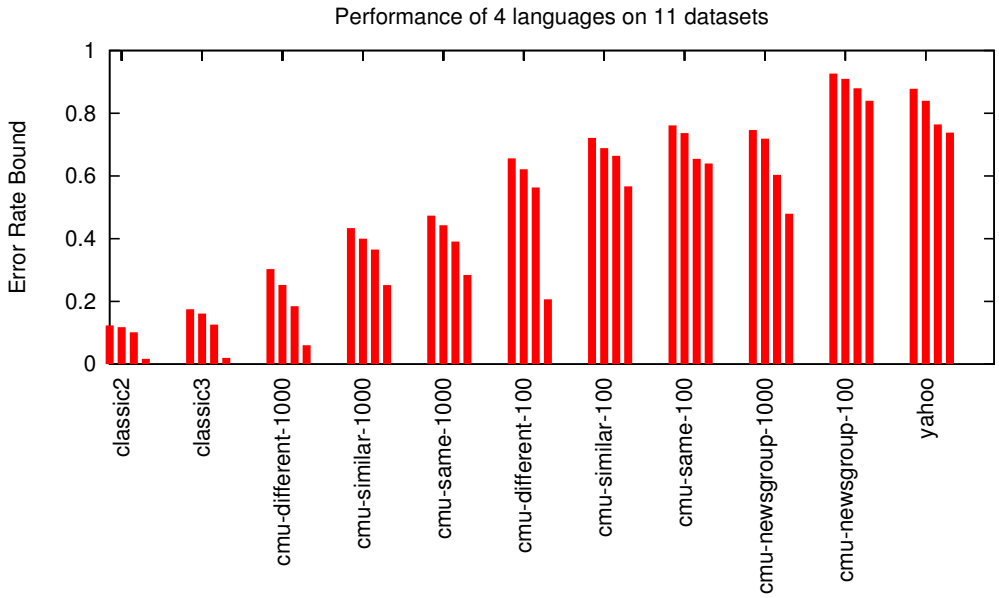


Figure 7.6: Test-set error-rate bounds on 11 datasets for 4 languages: Simple, Init, Cluster, and Algo.

Cluster, since Cluster takes into account the optimization over all number of clusters while Init does not. To make the comparison fair, we compare the average bound of Init to that for Cluster. Similar arguments apply to other languages. Figure 7.6 displays comparisons of the 4 languages on all the datasets under consideration. As shown in Figure 7.6, there seems to be some advantage to using a more complicated language, i.e., trying to optimize over several iterations, cluster numbers and algorithms. In practice, using a more complicated language of course implies more computational effort.

## 7.5 Discussion

The PAC-MDL bound provides an objective criterion for evaluation, comparison and model selection for clustering that is applicable when the goal of clustering is related to prediction. Experimental results show that this criterion is practically

and flexibly useful.

It is particularly striking (and perhaps even shocking) to notice test-set error-rate bounds achieved by the best clustering algorithms are very competitive with various supervised learning bounds on the true error rate of learned classifiers. This good performance suggests that clustering algorithms are doing something fundamentally "right" for prediction purposes on natural datasets.

# Chapter 8

# Rate Distortion with Bregman Divergences

In this chapter, we study the connection between Bregman clustering algorithms and lossy compression schemes. In particular, we focus on the relationship of our work with Shannon's rate distortion theory, showing connections between learning mixtures of exponential distributions, the Bregman soft clustering problem, and the rate distortion problem where distortion is measured using regular Bregman divergences [BDGM04]. Then, we show that all these problems involve a trade-off between compression and loss in Bregman information. The information bottleneck method emerges as a special case of this viewpoint for a particular choice of Bregman divergence. We restrict our attention to regular exponential families and regular Bregman divergences in this chapter.

[0]The work presented in this chapter has earlier appeared in part as [BDGM04].

## 8.1 Rate Distortion Theory for Bregman Divergences

The central problem in rate distortion theory [CT91] is to compute the rate distortion function $R(D)$, which is defined as the minimum achievable rate for a specified level of expected distortion $D$, and can be mathematically expressed as

$$R(D) = \min_{p(\hat{x}|x):E_{X,\hat{X}}[d(X,\hat{X})]\leq D} I(X;\hat{X}) \ , \tag{8.1}$$

where $I(X;\hat{X})$ is the mutual information of $X$ and $\hat{X}$.

Rose (1994) showed that the rate distortion problem for the squared Euclidean distortion and for any source whose support is a bounded set can be solved either analytically or through a numerical computation technique called the mapping approach [Ros94]. In this section, we generalize this result to all Bregman divergences. We start by presenting a new analytic lower bound on the rate distortion function for Bregman divergences, which we call the *Shannon-Bregman lower bound*.

**Theorem 9** *The rate distortion function for a source $X \sim p(x)$ and a Bregman divergence $d_\phi$ is always lower bounded by the* **Shannon-Bregman lower bound** *$R_L(D)$ defined as*

$$R_L(D) = H(X) + \sup_{\gamma\geq 0} \ \{-\gamma D + E_X[\log f_{\gamma\phi}(X)]\} \, ,$$

*where $H(X)$ denotes the (differential) entropy of $X$ and $f_{\gamma\phi}$ is the unique function that satisfies*

$$\int_{\mathrm{dom}(\phi)} \exp(-d_{\gamma\phi}(t,\mu)) f_{\gamma\phi}(t) \ dt = 1, \ \forall \mu \in \mathrm{dom}(\phi) \ .$$

The proof of the theorem is given in Appendix C. The Shannon-Bregman lower bound plays the same role for Bregman divergences as the *Shannon lower bound* for "difference" distortion measures [Ber71], i.e., distortions of the form $d(x, y) = \rho(x - y)$ for any non-negative function $\rho(\cdot)$. Rose (1994) used the Shannon lower bound for squared Euclidean distortion to reduce the rate distortion problem into two mutually exclusive solvable cases. More specifically, Rose (1994) showed that for squared Euclidean distortion and any source whose support is a bounded set, either (a) the rate-distortion function equals the Shannon lower bound, or (b) the optimal support of the reproduction random variable is finite, in which case the rate distortion function can be numerically computed using the mapping approach. Our following theorem states a significantly more general result.

**Theorem 10** *Consider the rate distortion problem for a source $X \sim p(x)$ and a Bregman divergence $d_\phi$. Let $\hat{\mathcal{X}}_s(D)$ be the support of the optimal reproduction random variable for an expected distortion $D$. If $\hat{\mathcal{X}}_s(D)$ contains an accumulation point, then $R(D) = R_L(D)$.*

The proof of the theorem is given in Appendix C. If $R(D) > R_L(D)$, then $\hat{\mathcal{X}}_s(D)$ does not contain an accumulation point. Further, if the source alphabet is a bounded set, $\hat{\mathcal{X}}_s(D)$ is a finite set using the Bolzano-Weierstrass theorem. Thus, the rate distortion problem for Bregman divergences and sources with bounded support can be divided into two cases, of which the first one can be solved analytically using the Shannon-Bregman lower bound and the second one requires a numerical solution involving a finite reproduction alphabet. Therefore, in the next section, we focus only on the second case. In fact, we solve a simpler problem assuming that the cardinality of the optimal support of the reproduction random variable is known. This assumption is reasonable since deterministic annealing methods [Ros98] can be applied to empirically determine the appropriate cardinality at any distortion value.

## 8.2 Rate Distortion for Fixed Finite Cardinality Reproduction Alphabet

In this section, we consider the *joint* problem of finding the optimal support $\hat{\mathcal{X}}_s$ of the reproduction random variable with $|\hat{\mathcal{X}}_s| = k$ as well as the optimal probabilistic assignments $p(\hat{x}|x)$ that achieve the rate-distortion function for a given source. When the distortion measure is a Bregman divergence, the problem can be formally stated as follows:

$$\min_{\substack{\hat{\mathcal{X}}_s,\ p(\hat{x}|x) \\ |\hat{\mathcal{X}}_s|=k}} \{I(X;\hat{X}) + \beta_D E_{X,\hat{X}}[d_\phi(X,\hat{X})], \tag{8.2}$$

where $\beta_D$ is the optimal Lagrange multiplier that depends on the chosen tolerance level $D$ of the expected distortion. We shall refer to problem (8.2) as rate distortion with a support of fixed finite cardinality (RDFC). It is important to note that unlike the original rate distortion problem (8.1), the RDFC problem is not a convex optimization problem, since it involves optimizing over both $\hat{\mathcal{X}}_s$ and $p(\hat{x}|x)$. Hence, it is difficult to obtain the globally optimal solution. However, since the minimization is over two sets of arguments, namely $p(\hat{x}|x)$ and $\hat{\mathcal{X}}_s$, the objective function in (8.2) can be greedily minimized by iteratively optimizing over the individual arguments yielding a solution that is locally optimal.

**Lemma 8 (Cover & Thomas, 1991)** [1] *The solution to the problem*

$$\min_{p(\hat{x}|x)} \{I(X;\hat{X}) + \beta_D E_{X,\hat{X}}[d_\phi(X,\hat{X})],$$

*for a fixed $\hat{\mathcal{X}}_s$ is given by*

$$p(\hat{x}|x) = \frac{p(\hat{x})}{N(x,\beta_D)} \exp(-\beta_D d_\phi(x,\hat{x})),$$

[1] Lemmas 8 and 9 hold irrespective of whether $X$ is a continuous or discrete random variable.

**Algorithm 10** Computation of Rate Distortion Curve for Bregman Divergences

**Input:** $X \sim p(x)$ over $x = \{x_i\}_{i=1}^n \subset \text{dom}(\phi) \subseteq \mathbb{R}^m$, Bregman divergence $d_\phi$, $k = |\hat{\mathcal{X}}_s|$, variational parameter $\beta$ corresponding to a point on $R(D)$ curve

**Output:** $\hat{\mathcal{X}}_s^* = \{\hat{x}_h\}_{h=1}^k$, $P^* = \{\{p(\hat{x}_h|x_i)\}_{h=1}^k\}_{i=1}^n$ that (locally) optimizes (8.2), rate-distortion trade-off $(R_\beta, D_\beta)$ at $\beta$.

**Method:**

Initialize with some $\{\hat{x}_h\}_{h=1}^k \subset \text{dom}(\phi)$
**repeat**
  {Blahut Arimoto Step ($p(\hat{x}|x)$ using Lemma 8)}
  **repeat**
    **for** $i = 1$ to $n$ **do**
      **for** $h = 1$ to $k$ **do**
        $p(\hat{x}_h|x_i) \leftarrow \frac{p(\hat{x}_h)}{N(x_i,\beta)} \exp(-\beta d_\phi(x_i, \hat{x}_h))$,
      **end for**
    **end for**
    **for** $h = 1$ to $k$ **do**
      $p(\hat{x}_h) \leftarrow \sum_{i=1}^n p(\hat{x}_h|x_i)p(x_i)$
    **end for**
  **until** *convergence*
  {Support Estimation Step ($\hat{\mathcal{X}}_s$ using Lemma 9)}
  **for** $h = 1$ to $k$ **do**
    $\hat{x}_h \leftarrow \sum_{i=1}^n p(x_i|\hat{x}_h)x_i$
  **end for**
**until** *convergence*
Compute $D_\beta = \sum_{x,\hat{x}} p(x)p(\hat{x}|x)d_\phi(x, \hat{x})$
Compute $R_\beta = \sum_{x,\hat{x}} p(x)p(\hat{x}|x) \log \frac{p(\hat{x}|x)}{p(\hat{x})}$

*where* $p(\hat{x}) = E_{X|\hat{x}}[p(X)]$ *and* $N(x, \beta_D)$ *is the partition function.*

**Lemma 9** *The solution to the problem,*

$$\min_{\hat{\mathcal{X}}_s} \{I(X; \hat{X}) + \beta_D E_{X,\hat{X}}[d_\phi(X, \hat{X})],$$

*for fixed probabilistic assignments* $p(\hat{x}|x)$ *is given by*

$$\hat{x}^* = E_{X|\hat{x}}[X].$$

Lemma 8 follows directly from the self-consistent equations for the solution of the rate distortion problem [CT91] while Lemma 9 follows from Proposition 1. Based on these results, we obtain an alternate minimization algorithm for computing the rate distortion function (Algorithm 10), guaranteed to achieve local optimality.

**Theorem 11** *The alternate minimization algorithm (Algorithm 10) for the RDFC problem (8.2) converges to a solution that is locally optimal, i.e., the objective function in (8.2) cannot be decreased by either changing $p(\hat{x}|x)$ or $\hat{\mathcal{X}}_s$.*

## 8.3 Equivalence with Mixture Estimation for Exponential Families

The maximum likelihood mixture estimation (MLME) problem involves finding the mixture of $k$ distributions from a specified parametric family $\mathcal{F}$ that best fits the observed data in terms of the log-likelihood. This problem seems different from the rate distortion problem since MLME only assumes knowledge of a finite set of independent samples of the random variable and not the actual distribution. However, as we shall show, it is equivalent to the rate distortion problem when the source distribution in the rate distortion setting equals the empirical distribution over the sampled data points.

The standard way to address the mixture estimation problem is to introduce a hidden random variable associated with the choice of mixture component. Let $\mathcal{X}_d$ be the finite set of independent samples corresponding to the observed random variable $X$. Let $\hat{X}$ be the hidden random variable corresponding to the choice of the mixture component and taking values in $\hat{\mathcal{X}}_s$ with $|\hat{\mathcal{X}}_s| = k$. The mixture distribution $p(x)$ can be viewed as the marginal induced from the joint distribution $p(x, \hat{x})$ such that each conditional distribution $p(x|\hat{x})$ belongs to the specified parametric family $\mathcal{F}$. The mixture estimation problem can be formally stated as the problem of maximizing

the average incomplete log-likelihood of the data, i.e., $\frac{1}{n}\sum_{x\in\mathcal{X}_d}\log p(x)$, over all mixture distributions $p(x)$ consisting of $k$ component distributions from $\mathcal{F}$. The MLME problem has been shown [NH98, Ros98] to be equivalent to the problem of minimizing the variational free energy of a statistical system, where the physical states correspond to the values of the unknown random variable $\hat{X}$ and the energy of each state is given by the negative joint log-likelihood ($-\log p(x,\hat{x})$). The negative of this variational free energy can be expressed as the sum of the entropy of the conditional distribution $p(\hat{x}|x)$ and the expected complete log-likelihood with respect to $p(x,\hat{x})$. Therefore, the minimum free energy problem and hence, the MLME problem can be expressed as

$$\min_{\substack{\hat{\mathcal{X}}_s,\ p(\hat{x}|x)\\ |\hat{\mathcal{X}}_s|=k}} \left\{-E_{X,\hat{X}}[\log p(X,\hat{X})] - H(\hat{X}|X)\right\}, \tag{8.3}$$

where $X\sim p_d(x)$, the empirical distribution over the sample set $\mathcal{X}_d$, the joint distribution $p(x,\hat{x}) = p(\hat{x})p(x|\hat{x})$ such that $p(x|\hat{x})\in\mathcal{F}$ and the minimization is performed over $\hat{\mathcal{X}}_s$ and $p(\hat{x}|x)$, which uniquely determine the mixture distribution $p(x)$.

Now, consider the case when the specified parametric family $\mathcal{F}$ is an exponential family $\mathcal{F}_\psi$ with a log-partition function $\psi$ so that $p_{(\psi,\theta)}(x)\in\mathcal{F}_\psi$ is given by

$$p_{(\psi,\theta)}(x) = \exp\left(\langle x,\theta\rangle - \psi(\theta)\right)\ ,$$

over some measure $\nu(x)$ where $\theta\in\mathrm{dom}(\psi)$ is the natural parameter. Although, the MLME problem (8.3) assumes that $\mathcal{F}_\psi$ is fully specified, typically, only a meta family $\mathcal{M}_\psi$ consisting of scaled versions of $\mathcal{F}_\psi$ is specified. To make this more precise, we define the scaled versions of $\mathcal{F}_\psi$ as the parametric families $\mathcal{F}_\psi^{(\beta)},\ \beta\geq 0$, such that $p^{(\beta)}_{(\psi,\theta)}(x)\in\mathcal{F}_\psi^{(\beta)}$ is given by

$$p^{(\beta)}_{(\psi,\theta)}(x) \quad\propto\quad (p_{(\psi,\theta)}(x))^\beta\ , \tag{8.4}$$

where $p_{(\psi,\theta)}(x) \in \mathcal{F}_\psi$ and $\mathcal{M}_\psi = \{\mathcal{F}_\psi^{(\beta)}, \beta \geq 0\}$. It can be shown that each $\mathcal{F}_\psi^{(\beta)}$ is itself an exponential family with a log-partition function $\psi_\beta(\theta) = \beta\psi(\theta/\beta)$. For example, the set of all unit variance Gaussian distributions over $\mathbb{R}$ is an exponential family $\mathcal{F}_\psi$ with $\psi(\theta) = \theta^2/2$. All *constant* variance Gaussian families are the scaled versions of this $\mathcal{F}_\psi$, and $\mathcal{M}_\psi$ is the set of all the scaled versions. To perform mixture modeling, we need to choose a particular member of the meta family $\mathcal{M}_\psi$, i.e., a particular value for the scaling factor $\beta$. Usually, $\beta$ is implicitly chosen to be 1 with $\mathcal{F}_\psi$ being a canonical representation of the meta family $\mathcal{M}_\psi$. In practice, appropriate choice of $\beta$ has led to improved results on natural datasets, e.g., see [Nig01] for scaled families on the mixture of multinomials model.

Using (8.4) in (8.3), we note that the scaling factor $\beta$ determines the relative importance of expected complete log-likelihood and the assignment entropy terms in the maximum likelihood problem (8.3), and consequently, the degree of "softness" in the assignments $p(\hat{x}|x)$. In particular, the assignment entropy term $H(\hat{X}|X)$ is significant for low $\beta$ leading to an almost uniform assignment, whereas for high $\beta$, the entropy term becomes insignificant resulting in hard assignments between $X$ and $\hat{X}$. It is, therefore, important to choose $\beta$ appropriately based on the desired accuracy and softness constraints. We present an information theoretic analysis for making this choice by demonstrating an equivalence between the RDFC problem for a specified distortion constraint and the MLME problem based on a particular member of a meta exponential family with scaling factor $\beta$ that depends on $D$.

### 8.3.1 Equivalence Theorem

Recall Theorem 7 that establishes a bijection between regular Bregman divergences and regular exponential families. Then, the Bregman divergence $d_\phi(\mathbf{x}, \mu)$ corresponds to the exponential density $p_{(\psi,\theta)}(x) \in \mathcal{F}_\psi$. Based on the bijection theorem, the conditional distribution $p_{(\psi,\hat{x})}(x) \in \mathcal{F}_\psi$ is given by $p_{(\psi,\hat{\mathbf{z}})}(x) = \exp(-d_\phi(x, \hat{x}))$,

where $\hat{x}$ is the expectation parameter, $\phi$ is the Legendre conjugate of $\psi$ and $d_\phi$ is the Bregman divergence derived from $\phi$. Hence, the Bregman divergence $d_\phi(x, \hat{x})$ is related to the negative log-likelihood $(-\log p_{(\psi,\hat{x})}(x))$ of the corresponding exponential distribution. We use this observation to prove the following equivalence.

**Theorem 12** *Consider a source $X \sim p_d(x)$. Then, the RDFC problem (8.2) for $X$ with Bregman distortion $d_\phi$, tolerable expected distortion $D$ with $|\hat{\mathcal{X}}_s| = k$ is equivalent to the MLME problem (8.3) for a mixture model with $k$ distributions from the scaled exponential family $\mathcal{F}_\psi^{(\beta_D)}$, where $\beta_D$ is the optimal Lagrange multiplier for the RDFC problem and $\psi$ is the Legendre conjugate of $\phi$.*

*Proof:* It is sufficient to compare the objective functions of the problems (8.2) and (8.3) as both are minimization problems with identical arguments and constraints. For the RDFC problem (8.2) based on Bregman divergence $d_\phi$ and tolerable level of distortion $D$, the objective function is given by

$$\begin{aligned} J_{RDFC}(\hat{\mathcal{X}}_s, p(\hat{x}|x)) &= I(X;\hat{X}) + \beta_D E_{X,\hat{X}}[d_\phi(X,\hat{X})] \\ &= E_{X,\hat{X}}[\log p(\hat{X}|X) - \log p(\hat{X}) + \beta_D d_\phi(X,\hat{X})] \ . \end{aligned}$$

Since $\psi$ is the conjugate of $\phi$, the exponential family $\mathcal{F}_\psi$ corresponding to the Bregman divergence $d_\phi$ is given by (4.11) and the scaled version $\mathcal{F}_\psi^{(\beta_D)}$ is obtained using (8.4). Hence, the objective function of the MLME problem (8.3) based on the exponential family $\mathcal{F}_\psi^{(\beta_D)}$ is given by

$$\begin{aligned} J_{MLME}(\hat{\mathcal{X}}_s, p(\hat{x}|x)) &= -E_{X,\hat{X}}[\log p(X,\hat{X})] - H(\hat{X}|X) \\ &= E_{X,\hat{X}}[\log p(\hat{X}|X) - \log p(\hat{X}) + \beta_D d_\phi(X,\hat{X})] \ . \end{aligned}$$

The objective functions $J_{MLME}$ and $J_{RDFC}$ are exactly same and hence, the equivalence follows. ■

The equivalence theorem gives an information theoretic recipe for choosing the appropriate scaled exponential family for a mixture modeling based on the desired model accuracy constraints. In particular, the Bregman distortion constraint $E_{X,\hat{X}}[d_\phi(X,\hat{X})] \leq D$, which is equivalent to a conditional entropy constraint $H(X|\hat{X}) \leq D$, specifies the desired level of model accuracy. Then, the appropriate exponential family for mixture modeling is $\mathcal{F}_\psi^{(\beta_D)}$ where $\beta_D$ is the optimal Lagrange multiplier of the RDFC problem. This follows since the optimal solution $(p(\hat{x}|x), \hat{\mathcal{X}}_s)$ of the RDFC problem exactly satisfies the condition $p(x|\hat{x}) \in \mathcal{F}_\psi^{(\beta_D)}$.

From Theorem 12, the objective function of the MLME problem corresponding to $F_\psi^{(\beta)}$ can be written as $I(X,\hat{X}) + \beta E_{X,\hat{X}}[d_\phi(X,\hat{X})]$. Therefore, solving the MLME problem based on any exponential family $\mathcal{F}_\psi^{(\beta)}$, $\beta \leq \beta_D$ such that $E_{X,\hat{X}}[d_\phi(X,\hat{X})] \leq D$ yields a solution identical to that of the unconstrained MLME problem based on $\mathcal{F}_\psi^{(\beta_D)}$, and the equivalent RDFC problem. In particular, the constrained MLME problem based on $\mathcal{F}_\psi \equiv \mathcal{F}_\psi^{(1)}$ such that $E_{X,\hat{X}}[d_\phi(X,\hat{X})] \leq D$ is equivalent to RDFC problem for all $D$ such that $\beta_D \geq 1$. Further, the constrained MLME problem based on $F_\psi^{(0^+)}$ is equivalent to the RDFC problem for all $D$.

The equivalence also suggests that that the RDFC problem can also be solved by the expectation maximization (EM) algorithm [RW84]. The update equations in both the algorithms are identical, the only difference being the order in which they are executed, i.e., the two algorithms correspond to two different ways of cyclic minimization. Both the algorithms are guaranteed to converge to a locally optimal solution, but the actual solutions could be different. In fact, any algorithm that alternates between the three updates, viz, $p(\hat{x}|x), p(\hat{x})$ and $\hat{x}$, will have similar guarantees. However, this class of algorithms have two drawbacks. First, the algorithms assume that the optimal cardinality $k(D)$ of the reproduction alphabet or the mix-

ture model for a given tolerable distortion $D$ is known though it is not the case in practice. Secondly, the algorithms are guaranteed to provide only locally optimal solutions. A practical technique that addresses these deficiencies is the deterministic annealing approach that starts with a high value of $D$ (i.e., a low positive $\beta_D$), where the optimal support of the reproduction random variable and the mixture model have cardinality one, and slowly decreases the tolerable distortion level $D$ (i.e., increases $\beta_D$) while detecting the phase transitions corresponding to changes in the cardinality. For details, see Rose (1998).

### 8.3.2 Equivalence with Soft Clustering

From Chapter 4, we know that the maximum likelihood mixture estimation problem for any regular exponential family is equivalent to the Bregman soft clustering problem for the corresponding regular Bregman divergence. Using this in conjunction with Theorem 12, we obtain the following equivalence relation between the RDFC problem and the Bregman soft clustering problem.

**Theorem 13** *Consider a source $X \sim p(x)$, where $p(x)$ is the empirical distribution over the samples. Then, the RDFC problem (8.2) for $X$ with regular Bregman distortion $d_\phi$, variational parameter $\beta$ and support of reproduction random variable assumed to have cardinality $k$ is equivalent to the Bregman soft clustering problem (4.13) based on the Bregman divergence $d_{\beta\phi}$ with number of clusters set to $k$.*

From the above theorem, it follows that Algorithm 3 can be used to solve the RDFC problem. Note that the update steps for $p(h|x)$ and $\pi_h$ in Algorithm 3 exactly correspond to the updates of $p(\hat{x}|x)$ and $p(\hat{x})$ in the Blahut-Arimoto step in Algorithm 10 for solving the RDFC problem. The update of $\mu_h$ in Algorithm 3 is equivalent to the update of $\hat{x}$ in the support estimation step in Algorithm 10. From the viewpoint of alternate minimization, the order of the three updates $p(\hat{x}|x), p(\hat{x})$ and $\hat{x}$ is interchangeable and does not affect the local optimality guarantees.

The Bregman soft clustering problem corresponds to the RDFC problem and not to the basic rate distortion problem (8.1). However, as mentioned earlier, both the problems yield the same solution for the rate distortion function when the optimal support set $|\hat{\mathcal{X}}_s|$ is finite and $k$ is suitably large. The solution is the rate distortion function and refers to the asymptotic rate [CT91] that can be achieved for a given distortion, when we are allowed to code the source symbols in blocks of size $m$ with $m \to \infty$.

It is also possible to consider a related rate distortion problem where the source symbols are coded using blocks of size 1. The resultant rate distortion function is referred to as "scalar" or "order 1" rate distortion function $R_1(D)$ [GN98]. The problem is solved by performing hard assignments of the source symbols to the closest codebook members, which is similar to the assignment step in the Bregman hard clustering problem. In fact, the "order 1" or "1-shot" rate distortion problem, assuming a known finite cardinality of the optimal reproduction support set, turns out to be exactly equivalent to the Bregman hard clustering problem.

## 8.4 Compression vs. Bregman Information Trade-off

In this section, we provide an alternate view of the RDFC problem (and hence, Bregman soft clustering) as a lossy compression problem where the objective is to balance the trade-off between compression and preservation of Bregman information. Intuitively, the reproduction random variable $\hat{X}$ is a coarser representation of the source random variable $X$ with less "information" than $X$. In rate distortion theory, the loss in "information" is quantified by the expected Bregman distortion between $X$ and $\hat{X}$. The following Theorem [BDGM04], which is along the same lines as Theorem 1, provides a direct way of quantifying the intuitive loss in "information" in terms of Bregman information.

**Theorem 14** *The expected Bregman distortion between the source and the reproduction random variables is exactly equal to the loss in Bregman information due to compression, i.e.,*

$$E_{p(x,\hat{x})}[d_\phi(X, \hat{X})] = I_\phi(X) - I_\phi(\hat{X}) \ ,$$

*where* $\hat{x} = E_{p(x|\hat{x})}[X]$.

*Proof:* To prove the above result, we first show that the expectations of the random variables $X$ and $\hat{X}$ are equal and use this to compute the loss in the Bregman information. Let $\mu_x$ and $\mu_{\hat{x}}$ be the expectations of $X$ and $\hat{X}$ respectively. Then,

$$\mu_{\hat{x}} = \sum_{\hat{x}} p(\hat{x})\hat{x} = \sum_{\hat{x}} p(\hat{x}) \sum_{x} p(x|\hat{x})x = \sum_{\hat{x},x} p(x, \hat{x})x = \sum_{x} p(x)x = \mu_x \ .$$

Assuming $\mu = \mu_x = \mu_{\hat{x}}$, the loss in the Bregman information is given by

$$\begin{aligned}
I_\phi(X) - I_\phi(\hat{X}) &= \sum_{x} p(x) d_\phi(x, \mu) - \sum_{\hat{x}} p(\hat{x}) d_\phi(\hat{x}, \mu) \\
&= \sum_{x,\hat{x}} p(\hat{x}, x)(d_\phi(x, \mu) - d_\phi(\hat{x}, \mu)) \\
&= \sum_{x,\hat{x}} p(\hat{x}, x)(\phi(x) - \phi(\mu) - \langle (x - \mu), \nabla\phi(\mu)\rangle \\
&\qquad\qquad - \phi(\hat{x}) + \phi(\mu) + \langle (\hat{x} - \mu), \nabla\phi(\mu)\rangle) \\
&= \sum_{x,\hat{x}} p(\hat{x}, x)(\phi(x) - \phi(\hat{x}) - (x - \hat{x}).\nabla\phi(\mu)) \\
&= \sum_{x,\hat{x}} p(\hat{x}, x)(\phi(x) - \phi(\hat{x})) \\
&= \sum_{x,\hat{x}} p(\hat{x}, x)(\phi(x) - \phi(\hat{x})) - \sum_{\hat{x}} p(\hat{x})\langle (\hat{x} - \hat{x}), \nabla\phi(\hat{x})\rangle \\
&= \sum_{x,\hat{x}} p(\hat{x}, x)(\phi(x) - \phi(\hat{x})) - \sum_{\hat{x}} p(\hat{x})\langle (\sum_{x} p(x|\hat{x})x - \hat{x}), \nabla\phi(\hat{x})\rangle \\
&= \sum_{x,\hat{x}} p(\hat{x}, x)(\phi(x) - \phi(\hat{x})) - \langle (x - \hat{x}), \nabla\phi(\hat{x})\rangle
\end{aligned}$$

$$= \sum_{x,\hat{x}} p(x,\hat{x}) d_\phi(x,\hat{x}) = E_{X,\hat{X}}[d_\phi(X,\hat{X})],$$

i.e., the expected distortion. That completes the proof. ■

The RDFC problem (8.2) can, therefore, be viewed as an optimization problem involving a trade-off between the mutual information $I(X;\hat{X})$ that measures the compression, and the loss in Bregman information $I_\phi(X) - I_\phi(\hat{X})$. Since the source random variable $X$ is known, the Bregman information $I_\phi(X)$ is fixed and minimizing the expected distortion is equivalent to maximizing the Bregman information of the compressed random variable $\hat{X}$. Hence, this constrained form of the RDFC problem (8.2) can be written as:

$$\min_{p(\hat{x}|x)} \{I(X;\hat{X}) - \beta I_\phi(\hat{X})\}, \tag{8.5}$$

where $\beta$ is the variational parameter corresponding to the desired point in the rate distortion curve and $\hat{x} = E_{X|\hat{x}}[X]$. The variational parameter $\beta$ determines the trade-off between the achieved compression and the preserved Bregman information.

### 8.4.1 Information Bottleneck Revisited

We now demonstrate how the information bottleneck method can be derived from the RDFC problem (8.5) for a suitable choice of Bregman divergence.

Let $Y \sim p(y),\ y \in \mathcal{Y}$ be a random variable. Let the sufficient statistic random vector $Z$ corresponding to a source $X$ be the conditional distribution of $Y$ given $X$, i.e., $Z = p(Y|X)$. $Z$ is just a concrete representation of the possibly abstract source $X$. Similarly, the random variable $\hat{Z} = p(Y|\hat{X})$ represents the reproduction random variable $\hat{X}$. This choice of sufficient statistic mapping is appropriate when the joint distribution of the random variables $X$ and $Y$ contains all the relevant information about $X$. For the above choice of sufficient statistic mapping, an ad-

ditional constraint that $\hat{Z}$ is the conditional expectation of $Z$ leads to the lossy compression problem (8.5) where we need to find the optimal assignments that balance the trade-off between compression and the loss in Bregman information. Now, from Example 5, the Bregman information $I_\phi(\hat{Z})$ of the random variable $\hat{Z}$ taking values over the set of conditional distributions $\{p(Y|\hat{x})\}$ with probability $p(\hat{x})$ is same as the mutual information $I(\hat{X};Y)$ of $\hat{X}$ and $Y$ when the Bregman divergence is the KL-divergence. Hence, the original problem (8.5) reduces to

$$\min_{p(\hat{x}|x)} \{I(X;\hat{X}) - \beta I(\hat{X};Y)\}, \tag{8.6}$$

since $p(\hat{z}|z) = p(\hat{x}|x)$ and $I(Z;\hat{Z}) = I(X;\hat{X})$, where $\beta$ is the variational parameter. This is identical to the information bottleneck (IB) formulation [TPB99]. Our framework reveals that the IB assumption that the mutual information with respect to another random variable $Y$ holds all the relevant information for comparing the different source entities is equivalent to assuming that (a) $P(Y|X)$ is the appropriate sufficient statistic representation and (b) the KL-divergence between the conditional distributions of $Y$ is the appropriate distortion measure. Further, the assumption about the conditional independence of $Y$ and $\hat{X}$ given $X$, i.e., the Markov chain condition $Y \leftrightarrow X \leftrightarrow \hat{X}$, is equivalent to the constraint that $\hat{Z}$ is the conditional expectation of $Z$, i.e., $\hat{z} = p(Y|\hat{x}) = E_{p(X|\hat{x})}[p(Y|X)] = E_{p(Z|\hat{z})}[Z]$.

Thus, the information bottleneck problem is a special case of the RDFC problem (8.2) and hence, also of the Bregman soft clustering problem and mixture estimation problem for exponential families. In particular, it is exactly equivalent to the mixture estimation problem based on the exponential family corresponding to KL-divergence, i.e., the multinomial family [CDS01]. Further, the iterative IB algorithm is the same as the EM algorithm for multinomial distributions, and also the Bregman soft clustering algorithm using KL-divergence, as has been previously shown in [SW02].

# Chapter 9

# Optimal Stochastic Prediction

Predicting the outcome of a random event based on partial information is an important problem in many domains. As we saw in chapter 2, when accuracy of prediction is measured by the squared loss, it is well known [Wil91] the conditional expectation is the optimal predictor of a random variable. If $X$ is the random variable to be predicted and $Z$ denotes the random variable for the available information, then

$$E[X|Z] = \underset{Y \in \sigma(Z)}{\operatorname{argmin}} \; E[\|X - Y\|^2] \; , \tag{9.1}$$

where $\sigma(Z)$ denotes the $\sigma$-algebra generated by $Z$.

A question arises naturally: *Are there other loss functions $F$ for which $E[X|Z]$ is the unique best predictor?* Some simple counter-examples lead to the general conviction that the existence of such loss functions would be rare and would have to possess very special properties. For example, if one uses the absolute error loss function ([LC98], Section 1.7) then any constant $a$ satisfying $P(X \leq a) \geq 1/2 \leq P(X \geq a)$, i.e., the median of $X$ and not $E[X]$, proves to be the best constant predictor. Recently [Ath99] studied the case of general convex loss functions and

[0]The work presented in this paper has earlier appeared in part as [BGW04].

obtained a criterion for which a best constant predictor exists.

In this chapter, we provide necessary and sufficient conditions for general loss functions under which the conditional expectation is the unique optimal predictor. First, we show that the optimality property of the conditional expectation holds for *all* Bregman Bregman Divergences. Since Bregman divergences are used as a loss function for prediction, we shall refer to the loss functions as Bregman Loss Functions (BLFs). Secondly, we show that the class of BLFs is exhaustive under mild conditions, i.e., if $\operatorname{argmin}_{y\in\mathbb{R}^d} E[F(X,y)] = E[X]$ for every random variable $X$, then the loss function $F$ has to be a BLF, up to an additive constant.

Note that since a differentiable convex function is necessarily continuously differentiable [Roc70, Theorem 25.5], the function $d_\phi$ is continuous. Moreover, if we write $\nabla_x$ as the gradient with respect to $x$, then the function

$$\nabla_x d_\phi(x,y) = \nabla\phi(x) - \nabla\phi(y)$$

is also continuous. Further, note that if $\mathcal{G}$ is a sub-$\sigma$-algebra of $\mathcal{F}$ and $E[X|\mathcal{G}]$ denotes the conditional expectation, all the results presented in this chapter remain true if one replaces $E[\cdot|Z]$ by $E[\cdot|\mathcal{G}]$, and simultaneously replacing functions of $Z$ by $\mathcal{G}$-measurable random variables.

## 9.1 The optimal Bregman predictor

In this section we will show that the conditional expectation is the unique optimal predictor for all BLFs, and that any nearly optimal predictor will converge in probability to the conditional expectation.

**Theorem 15 (Optimality Property)** *Let $\phi : \mathbb{R}^d \mapsto \mathbb{R}$ be a strictly convex, differentiable function and let $d_\phi$ be the corresponding BLF. Let $X$ be an arbitrary random variable taking values in $\mathbb{R}^d$ for which both $E[X]$ and $E[\phi(X)]$ are finite.*

*Then, among all functions of $Z$, the conditional expectation is the unique minimizer (up to a.s. equivalence) of the expected Bregman loss, i.e.,*

$$\text{argmin}_{Y \in \sigma(Z)} E[d_\phi(X, Y)] = E[X|Z].$$

*Proof:* Let $Y$ be any function of $Z$, and $Y^* \doteq E[X|Z]$. It follows from the definition that

$$\begin{aligned} & E[d_\phi(X, Y)] - \mathbf{E}[d_\phi(X, Y^*)] \\ = \;& E\left[\phi(Y^*) - \phi(Y) - \langle X - Y, \nabla\phi(Y)\rangle + \langle X - Y^*, \nabla\phi(Y^*)\rangle\right]. \end{aligned}$$

Meanwhile, for $Y$ being any function of $Z$ , we have

$$E[\langle X - Y, \nabla\phi(Y)\rangle] = E[E[\langle X - Y, \nabla\phi(Y)\rangle|Z]] = E[\langle Y^* - Y, \nabla\phi(Y)\rangle].$$

In particular, $E[\langle X - Y^*, \nabla\phi(Y^*)\rangle] = 0$. Therefore,

$$\begin{aligned} E[d_\phi(X, Y)] - \mathbf{E}[d_\phi(X, Y^*)] &= E[\phi(Y^*) - \phi(Y) - \langle Y^* - Y, \nabla\phi(Y)\rangle] \\ &= E[d_\phi(Y^*, Y)]. \end{aligned} \tag{9.2}$$

The theorem follows immediately from Property 1, Appendix A. ∎

**Theorem 16 (Convergence in Probability)** *In the setting of Theorem 15, if $\{Y_n\}$ is a sequence of functions of $Z$, such that*

$$E[d_\phi(X, Y_n)] \to E[d_\phi(X, Y^*)],$$

*where $Y^* \doteq \mathbf{E}[X|Z]$, then $Y_n \to Y^*$ in probability.*

*Proof:* It suffices to show that for any given $\epsilon, \delta > 0$, there exists a number $N$ such that

$$P(|Y_n - Y^*| \geq \delta) \leq \epsilon \ ,$$

$\forall n \geq N$. The integrability of $X$ (and hence of $Y^*$) suggests that for a given $\epsilon > 0, \exists M$ such that

$$P(|Y^*| \geq M) \leq \epsilon/2.$$

Hence

$$\begin{aligned} P(|Y_n - Y^*| \geq \delta) &\leq P(|Y_n - Y^*| \geq \delta, |Y^*| \leq M) + P(|Y^*| \geq M) \\ &\leq P(|Y_n - Y^*| \geq \delta, |Y^*| \leq M) + \epsilon/2. \end{aligned}$$

For every $x \in \mathbb{R}^d$, if we define

$$h(x) \doteq \inf\{d_\phi(x, y) : \ y \in \mathbb{R}^d, \ |y - x| \geq \delta\},$$

then the strict convexity of $\phi$ implies that $h(x) > 0, \forall x \in \mathbb{R}^d$, and

$$h(x) = \inf\{d_\phi(x, y) : \ y \in \mathbb{R}^d, \ |y - x| = \delta\}.$$

Since $d_\phi$ is continuous, the infimum is always achieved. Moreover, it can be shown that

$$\alpha \doteq \inf\{h(x) : |x| \leq M\} > 0. \tag{9.3}$$

For now assuming (9.3) to be true, we have

$$P(|Y_n - Y^*| \geq \delta) \leq P(d_\phi(Y^*, Y_n) \geq \alpha) + \epsilon/2 \leq \mathbf{E}[d_\phi(Y^*, Y_n)]/\alpha + \epsilon/2.$$

From the assumption on $\{Y_n\}$ and (9.2) it follows that $E[d_\phi(Y^*, Y_n)] \to 0$. Hence, there exists $N$ such that for $n \geq N$, $E[d_\phi(Y, Y_n)] \leq \epsilon\alpha/2$. Therefore, for $n \geq N$,

$$P(|Y_n - Y^*| \geq \delta) \leq \epsilon,$$

and hence we have convergence in probability.

Finally, we show that $\alpha > 0$. This is proved by contradiction. Clearly $\alpha \not< 0$. Suppose $\alpha = 0$. Then there exists a sequence $\{x_n\}$ with $|x_n| \leq M$ and a sequence $\{y_n\}$ with $|y_n - x_n| = \delta$ such that

$$h(x_n) = d_\phi(x_n, y_n) \to 0.$$

Since $\{x_n\}$ and $\{y_n\}$ are both bounded, there exists a subsequence (still indexed by $n$) such that

$$x_n \to \bar{x}, \quad y_n \to \bar{y}.$$

Clearly $|\bar{x}| \leq M$ and $|\bar{y} - \bar{x}| = \delta$. The continuity of $d_\phi$ yields that $d_\phi(\bar{x}, \bar{y}) = 0$, which contradicts $h(\bar{x}) > 0$. This completes the proof. ■

**Remark 1** Other types of convergence results may be obtained by imposing proper conditions on the function $\phi$. For example, it is easy to see that $Y_n \to Y^*$ in $\mathbb{L}^2$ if the Hessian matrix of $\phi$ is uniformly positive definite over $\mathbb{R}^d$ (in the 1-dim case, it amounts to $\inf_{x\in\mathbb{R}} \phi''(x) > 0$).

## 9.2 The Exhaustiveness property of BLFs

In this section we establish exhaustiveness results for the class of loss functions for which the conditional expectation is the optimal predictor. More precisely, under mild regularity conditions we show that for a non-negative loss function $F : \mathbb{R}^d \times$

$\mathbb{R}^d \mapsto \mathbb{R}$ if $\forall X, Z$,

$$\text{argmin}_{Y \in \sigma(Z)} \; E\left[F(X,Y)\right] = E[X|Z] \; , \tag{9.4}$$

then $F$ is a BLF.

**Remark 2** Indeed, we will prove the following slightly stronger result: A non-negative loss function $F$ has to be a BLF if

$$\text{argmin}_{y \in \mathbb{R}^d} \; E\left[F(X,y)\right] = E[X], \tag{9.5}$$

for every random variable $X$. In other words, if the expectation $E[X]$ is the best *constant* predictor for every random variable $X$, then $F$ is a BLF.

We will present the results separately for the one-dimensional (Theorem 17) and the higher-dimensional (Theorem 18) case, since the latter needs slightly stronger regularity conditions.

For ease of exposition, and without loss of generality, we will assume in Theorem 17 and Theorem 18 that $F(x,x) = 0, \forall x$. Indeed, if $F$ is a loss function satisfying (9.4), so is $\bar{F}(x,y) \doteq F(x,y) - F(x,x)$ with $\bar{F}(x,x) \equiv 0$.

**Theorem 17 ($d = 1$)** *Let $F : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ be a non-negative function such that $F(x,x) = 0, \forall x \in \mathbb{R}$. Assume that $F$ and $F_x$ are both continuous functions. If for all random variables $X$, $E[X]$ is the unique minimizer of $E[F(X,y)]$ over all constants $y \in \mathbb{R}^d$, i.e.,*

$$\text{argmin}_{y \in \mathbb{R}^d} \; E[F(X,y)] = E[X],$$

*then $F(x,y) = d_\phi(x,y)$ for some strictly convex, differentiable function $\phi : \mathbb{R} \mapsto \mathbb{R}$.*

*Proof:* The proof will be completed in three steps. First, we prove that $F = d_\phi$ for some convex, differentiable function $\phi$, under an additional assumption that

$F_y$ is continuous; we then extend this result to the general case by a mollification argument; finally, we show that $\phi$ is strictly convex.

**Step 1:** Assume $F_x$ and $F_y$ are both continuous. Fix arbitrarily $a, b \in \mathbb{R}$, and $p \in [0,1]$. Consider a random variable $X$ such that $P(X = a) = p$ and $P(X = b) = q$ with $p + q = 1$. Then from the assumption

$$pF(a,y) + qF(b,y) = E[F(X,y)] \geq E[F(X, E[X])] = pF(a, pa+qb) + qF(b, pa+qb)$$

for all $y \in R$. Moreover, if we consider the left-hand-side as a function of $y$, it equals the right-hand-side at $y = y^* \doteq E[X] = pa + qb$. Therefore, we must have

$$pF_y(a, y^*) + qF_y(b, y^*) = 0. \tag{9.6}$$

Substituting $p = (y^* - b)/(a - b)$ and rearranging terms yield

$$F_y(a, y^*)/(y^* - a) = F_y(b, y^*)/(y^* - b).$$

Since $a, b$ and $p$ are arbitrary, the above equality implies that the function

$$F_y(x,y)/(y-x)$$

is independent of $x$. Thus one can write, for some function $H$,

$$F_y(x,y) = (y-x)H(y), \tag{9.7}$$

where $H$ is continuous. Now define function $\phi$ by

$$\phi(y) \doteq \int_0^y \int_0^t H(s)dsdt.$$

Then $\phi$ is differentiable with $\phi(0) = \phi'(0) = 0$, $\phi''(y) = H(y)$. Since $F(x, x) = 0$, integration by parts for (9.7) leads to

$$F(x, y) = \int_x^y (s - x)H(s)\, ds = \phi(x) - \phi(y) - \phi'(y)(x - y).$$

It follows from the non-negativity of $F$ that $\phi$ is a convex function.

**Step 2:** Now we show that there exists a convex function $\phi$ such that $F = d_\phi$ under the assumption of the theorem. Consider a sequence of mollifiers, i.e., a sequence of functions $\{g_n\}$ defined on $\mathbb{R}$, which are non-negative, $\mathcal{C}^\infty$ and with compact support such that

$$\int_{\mathbb{R}} g_n(x)\, dx = 1.$$

A classical example for such a sequence of mollifiers is as follows: let

$$g(x) \doteq \begin{cases} c \exp\left\{1/(x^2 - 1)\right\} & \text{if } |x| < 1, \\ 0 & \text{if } |x| \geq 1, \end{cases}$$

where the constant $c$ is to be chosen so that $\int_{\mathbb{R}} g(x)dx = 1$, and define $g_n(x) \doteq ng(nx)$. The mollified version of $F$ is then given by

$$F_n(x, y) \doteq \int_{\mathbb{R}} F(x - u, y - u)g_n(u)\, du = \int_{\mathbb{R}} F(x - y + u, u)g_n(y - u)\, du.$$

It is standard to show that [GT01, Section 7.2] $F_n$ is continuously differentiable with respect to $x$ and $y$, and that

$$\lim_{n\to\infty} F_n(x, y) = F(x, y),$$

for every $x, y \in \mathbb{R}$.

Furthermore, it is easy to see that $F_n$ has the same property as $F$, i.e., $E[X]$ is the minimizer for the loss function $F_n$. Therefore, by the proof in Step 1, there exists a convex, differentiable function $\phi_n$ such that $\phi_n(0) = \phi_n'(0) = 0$ and

$$F_n(x, y) = \phi_n(x) - \phi_n(y) - \phi_n'(y)(x - y). \tag{9.8}$$

In particular, $F_n(x, 0) = \phi_n(x)$. Since $F_n(x, 0) \to F(x, 0)$ for every $x$, we have

$$\lim_{n\to\infty} \phi_n(x) = F(x, 0) \doteq \phi(x)$$

for every $x$. Since $\phi_n$'s are convex, so is their limit $\phi$. In particular, $\phi$ is continuous [Roc70, Theorem 10.1]. Setting $x = y + 1$ in equation (9.8), we have

$$\begin{aligned} \phi_n'(y) &= F_n(y + 1, y) - \phi_n(y + 1) + \phi_n(y) \\ \Rightarrow \lim_{n\to\infty} \phi_n'(y) &= F(y + 1, y) - \phi(y + 1) + \phi(y) \doteq f(y). \end{aligned}$$

Clearly $f$ is continuous. Letting $n \to \infty$ in both sides of equation (9.8), we have

$$F(x, y) = \phi(x) - \phi(y) - f(y)(x - y),$$

where $\phi$ is continuously differentiable, since $F$ is continuously differentiable with respect to $x$. Furthermore, the non-negativity of $F$ implies that $f(y)$ is a subgradient of $\phi$ [Roc70, Page 214]. Finally, the differentiability of $\phi$ suggests that its subdifferential is just its derivative [Roc70, Theorem 25.1]. It follows that $\phi'(y) = f(y)$, and hence $F = d_\phi$.

**Step 3:** It remains to show that $\phi$ is strictly convex. From step 2, we already know that $\phi$ is a convex function. We prove by contradiction that if $\phi$ is not strictly convex, the assumption of uniqueness will be violated. Suppose $\phi$ is not strictly convex. Then there exists an interval $I = [\ell_1, \ell_2]$ such that $\ell_1 < \ell_2$ and $\phi'(y) = \phi'(\ell_1)$ for

all $y \in I$. Consider a random variable $X$ such that $P(X = \ell_1) = P(X = \ell_2) = 1/2$. It is not difficult to check that any $y \in I$ is a minimizer. Indeed, $E[d_\phi(X, y)] \equiv 0$ for all $y \in I$. This is a contradiction, and we complete the proof. ∎

**Theorem 18** ($d \geq 2$) *Let $F : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ be a non-negative function such that $F(x, x) = 0, \forall x \in \mathbb{R}^d$. Assume that $F(x, y)$ and $F_{x_i x_j}(x, y)$, $1 \leq i, j \leq d$ are all continuous. For all random variables $X$ taking value in $\mathbb{R}^d$, if $E[X]$ is the unique minimizer of $E[F(X, y)]$ over all constants $y \in \mathbb{R}^d$, i.e.,*

$$\text{argmin}_{y \in \mathbb{R}^d} \; E[F(X, y)] = E[X],$$

*then $F(x, y) = d_\phi(x, y)$ for some strictly convex and differentiable function $\phi : \mathbb{R}^d \mapsto \mathbb{R}$.*

The proof is divided into three analogous steps as those in Theorem 17. The only essential difference is in Step 1, which relies on the following lemma. The lemma itself is a direct consequence of the celebrated Poincaré Lemma.

**Lemma 10** *Given a collection of continuous functions $\{h_{ij} : 1 \leq i, j \leq d\}$ defined on an open, convex set $U \subseteq \mathbb{R}^d$ ($d \geq 2$). If for all triples of indices $1 \leq i, j, k \leq d$,*

$$h_{ij} \equiv h_{ji}, \qquad \frac{\partial h_{ij}}{\partial x_k} \equiv \frac{\partial h_{kj}}{\partial x_i}.$$

*Then there exists a function $\Phi : U \mapsto \mathbb{R}$ such that $\Phi_{x_i x_j} = h_{ij}$.*

*Proof:* (of Lemma 10) We first show that there exists a sequence of functions $\{\phi_i : 1 \leq i \leq d\}$ defined on $U$ such that, for every index $i$,

$$\nabla \phi_i \equiv (h_{i1}, \ldots, h_{id})^T. \tag{9.9}$$

This follows from the given property for triplets of indices in conjunction with the Poincaré Lemma [Edw73, Theorem 8.1] applied to 1-forms, noting that every convex set is star-convex. It remains to show that there exists a function $\Phi$ such that

$$\nabla\Phi = (\phi_1, \ldots, \phi_d)^T.$$

Note that for any pair of indices $i, j$, from equation (9.9) and the given property, we have

$$\frac{\partial \phi_i}{\partial x_j} = h_{ij} = h_{ji} = \frac{\partial \phi_j}{\partial x_i}.$$

The existence of $\Phi$ now follows via the Poincaré Lemma. ∎

*Proof:* (of Theorem 18) **Step 1:** Assume that $F_{x_i x_j}$, $F_{x_i y_j}$ and $F_{y_i y_j}$, $1 \leq i, j \leq d$ are all continuous (i.e., $F$ is twice continuously differentiable). Fix arbitrarily $a, b \in \mathbb{R}^d$, and $p \in [0, 1]$. Consider a random variable $X$ such that $P(X = a) = p$ and $P(X = b) = q$ with $p + q = 1$. Similar to the proof of equation (9.6), we have

$$pF_{y_i}(a, y^*) + qF_{y_i}(b, y^*) = 0, \qquad \forall\ i = 1, \ldots, d,$$

at $y^* = pa + qb$. Taking derivatives over $p$ on both sides of the above equation and recalling $q = 1 - p$, we arrive at

$$F_{y_i}(a, y^*) - F_{y_i}(b, y^*) + \sum_{j=1}^{d} \left[pF_{y_i y_j}(a, y^*) + qF_{y_i y_j}(b, y^*)\right](a_j - b_j) = 0,$$

for every $i = 1, \ldots, d$. In particular, setting $p = 1$ leads to

$$F_{y_i}(a, a) - F_{y_i}(b, a) + \sum_{j=1}^{d} F_{y_i y_j}(a, a)(a_j - b_j) = 0, \qquad \forall\ i = 1, \ldots, d.$$

Because $F$ is non-negative and $F(x, x) \equiv 0$, we have $F_{y_i}(a, a) \equiv 0$. Writing $H_{ij}(a) \doteq F_{y_i y_j}(a, a)$, and noting that $a$ and $b$ are arbitrary, we may rewrite the the above

equation as

$$F_{y_i}(x,y) = \sum_{j=1}^{d} H_{ij}(y)(y_j - x_j), \qquad \forall\ x, y \in \mathbb{R}^d. \tag{9.10}$$

Since $F_{y_i}$ is continuously differentiable for every $i$, it follows easily that $H_{ij}$ is also continuously differentiable for all $1 \leq i, j \leq d$. We now claim that there exists a function $\phi : y \in \mathbb{R}^d \mapsto H(y) \in \mathbb{R}$ such that

$$\phi_{y_i y_j}(y) = H_{ij}(y), \qquad 1 \leq i, j \leq d. \tag{9.11}$$

Indeed, from equation (9.10), we see that for every $k = 1, \ldots, d$,

$$F_{y_i y_k}(x,y) = \sum_{j=1}^{d} (H_{ij})_{y_k}(y)(y_j - x_j) + H_{ik}(y),$$

and

$$F_{y_k y_i}(x,y) = \sum_{j=1}^{d} (H_{kj})_{y_i}(y)(y_j - x_j) + H_{ki}(y),$$

Now, $F_{y_i y_k} = F_{y_k y_i}$ implies

$$H_{ik} \equiv H_{ki}, \qquad (H_{ij})_{y_k} \equiv (H_{kj})_{y_i}. \tag{9.12}$$

The existence of $\phi$ now follows from Lemma 10.

Now, from equation (9.10) we have

$$F_{y_i}(x,y) = \sum_{j=1}^{d} \phi_{y_i y_j}(y)(y_j - x_j) = \frac{\partial}{\partial y_i} \left[ -\phi(y) - \langle \nabla\phi(y), x - y \rangle \right],$$

which, combined with the condition $F(x,x) \equiv 0$, readily yields

$$F(x,y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y \rangle = d_\phi(x,y).$$

The convexity of $\phi$ is implied by the non-negativity of $F$.

**Step 2** and **Step 3:** Now repeating the same steps as those in the proof of Theorem 17, Theorem 18 is immediate. ∎

## 9.3 Discussion

Throughout the chapter, for the purpose of concise presentation, we assume that the convex function $\phi$ is finite on the whole Euclidean space $\mathbb{R}^d$, and the random variable $X$ is allowed to take values in the whole $\mathbb{R}^d$. However, the same methodology with very minor modifications will lead to similar results when $\mathbb{R}^d$ is replaced by an open convex subset of $\mathbb{R}^d$, typically $\mathrm{dom}(\phi)$, the domain of $\phi$.

Loss functions that lead to the optimality of the conditional expectation have been studied in earlier literature. An analysis applicable to difference distortion measures, i.e., distortions of the form $F(x, y) = C(x - y)$, is presented in [Tre68]. In particular, it is shown that if $C$ is symmetric, i.e., $C(z) = C(-z)$, and strictly convex, and the conditional probability density of $X$ given $Z$ is symmetric around the conditional expectation $E[X|Z]$, then the conditional expectation is the best predictor. Note that the third assumption regarding the symmetry of the conditional probability distribution of $X$ given $\sigma(Z)$ is very strong and makes the optimality result very restricted. The results discussed in this chapter apply to all random variables and hence are direct generalizations of the well-known least-squares prediction results [KT74, Wil91].

BLFs have been extensively studied in the context of convex optimization and related problems; see [CZ98, Csi91] and reference therein. Ben-Tal et. al. [BTCT89] applied one-dimensional BLFs to the analysis of entropic means, where the authors studied a different optimization problem, namely $\min_{x\in\mathbb{R}} E[F(x, Y)]$ for a fixed random variable $Y$. An axiomatic characterization of a wide class of distortion functions, including $f$-divergences [Csi67] and BLFs, was obtained in Csiszár's seminal

work [Csi91]. In another paper by Csiszár [Csi95], BLFs were used primarily for analyzing generalized projections for non-negative functions on convex sets. The interesting connection between reverse I-divergence (which is a special case of BLFs) and arithmetic mean was briefly mentioned in [Csi95] without further elaboration and study. Therefore, compared to Csiszár's work, we study BLFs from a different (i.e., probabilistic) perspective and with different methodologies. We provide a complete characterization of the fundamental relationship between BLFs and conditional expectation.

# Chapter 10

# Conclusion

In this thesis, we studied scalable clustering algorithms that are applicable to diverse problems, while taking into account one or more of the constraints that a practical problem may have. Also, we presented methods of evaluation and model selection, and analyzed connections of the proposed models to rate distortion theory and stochastic prediction.

We establish several fundamental theoretical results in this thesis:

1. In Chapter 3, we unify a large class of centroid based partitional clustering algorithms using Bregman divergences.

2. In Chapter 4, we establish a bijection between regular exponential family distributions and regular Bregman divergences. As a result, the log-likelihood of any exponential family distribution can always be written as the negative of a uniquely determined Bregman divergence. This leads to a class of efficient Bregman soft clustering algorithms.

3. In Chapter 8, we show that the rate distortion problem using Bregman divergences to measure distortion can be solved either analytically using the Shannon-Bregman lower bound, or computationally using the Bregman soft

clustering algorithm.

4. In Chapter 9, we show that the conditional expectation is the optimal predictor for a random variable if and only if the prediction error is measured by a Bregman divergence. This is a direct generalization of a widely known and utilized result regarding the least-mean-square predictor in probability theory, stochastic prediction and control.

In addition to the theoretical results, there are several practical contributions of this thesis:

1. In Chapter 5, a new class of algorithms are proposed and analyzed for clustering directional data. The algorithms can be applied to both the batch as well as streaming data. The proposed algorithms perform surprisingly well for high-dimensional text data. In fact, in a related work [BDGS03], we have extended the directional model even further and have reported some of the best results in text clustering on certain benchmarks. Preliminary experiments show that the algorithms are also effective in clustering micro-array gene-expression data [BDGS03] and hence potentially valuable for computational biology.

2. In Chapter 6, a general framework for scaling up balanced clustering algorithms is proposed. The simplicity of the proposed framework, its applicability to a wide variety of base clustering algorithms, along with the guarantees it gives makes the model practical for a large variety of real-life scenarios.

3. While evaluation and model selection remains a debated issue in clustering, in Chapter 7 we propose a clean and objective method using PAC-MDL bounds in a transductive setting. Interestingly, the same methodology for model selection can be utilized by other predictive models that do model selection using a validation set.

In addition to the specific practical and theoretical results, the thesis strongly suggests that there is a deep connection between unsupervised learning and lossy compression. It seems that the study of lossy compression from a predictive modeling viewpoint will give rise to better practical algorithms as well as theoretical understanding of clustering and related learning models.

# Appendix A

# Properties of Bregman Divergences

In this appendix, we list some well-known useful properties of Bregman Divergences. Let $\phi : \mathcal{S} \mapsto \mathbb{R}$ be a strictly convex, differentiable function defined on a convex set $\mathcal{S} = \text{dom}(\phi) \subseteq \mathbb{R}^d$ and let $d_\phi : \mathcal{S} \times \text{int}(\mathcal{S}) \mapsto [0, \infty)$ be its Bregman divergence, i.e, $d_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - \langle \mathbf{x} - \mathbf{y}, \nabla\phi(\mathbf{y}) \rangle$. Further, let $\psi : \Theta \mapsto \mathbb{R}$ be the conjugate function of $\phi$. Then, the following properties are true.

1. **Non-negativity:** $\forall \mathbf{x} \in \mathcal{S}, \mathbf{y} \in \text{int}(\mathcal{S}), d_\phi(\mathbf{x}, \mathbf{y}) \geq 0$ and equality holds if and only if $\mathbf{x} = \mathbf{y}$.

2. **Convexity:** $d_\phi$ is always convex in the first argument, but not necessarily convex in the second argument. Squared Euclidean distance and KL-divergence are examples of Bregman divergences that are convex in both their arguments, but the Bregman divergence corresponding to the strictly convex function $\phi(x) = x^3$, defined on $\mathbb{R}_+$, given by $d_\phi(x, y) = x^3 - y^3 - 3(x - y)y^2$ is not necessarily convex in $y$.

3. **Linearity:** Bregman divergence is a linear operator i.e., $\forall \mathbf{x} \in \mathcal{S}, \mathbf{y} \in \text{int}(\mathcal{S})$,

$$\begin{aligned} d_{\phi_1+\phi_2}(\mathbf{x},\mathbf{y}) &= d_{\phi_1}(\mathbf{x},\mathbf{y}) + d_{\phi_2}(\mathbf{x},\mathbf{y}) \ , \\ d_{c\phi}(\mathbf{x},\mathbf{y}) &= c d_{\phi}(\mathbf{x},\mathbf{y}) \quad (\text{for } c \geq 0) \ . \end{aligned}$$

4. **Equivalence classes:** The Bregman divergences of functions differing only in affine terms are identical i.e., if $\phi(\mathbf{x}) = \phi_0(\mathbf{x}) + \langle \mathbf{b}, \mathbf{x} \rangle + c$ where $\mathbf{b} \in \mathbb{R}^d$ and $c \in \mathbb{R}$, then $d_\phi(\mathbf{x},\mathbf{y}) = d_{\phi_0}(\mathbf{x},\mathbf{y}), \forall \mathbf{x} \in \mathcal{S}, \mathbf{y} \in \text{int}(\mathcal{S})$. Hence, the set of all strictly convex, differentiable functions on a domain $\mathcal{S}$ can be partitioned into equivalence classes of the form

$$[\phi_0] = \{\phi \ | d_\phi(\mathbf{x},\mathbf{y}) = d_{\phi_0}(\mathbf{x},\mathbf{y}) \ \forall \ \mathbf{x} \in \mathcal{S}, \ \mathbf{y} \in \text{int}(\mathcal{S})\}.$$

5. **Linear separator:** The locus of all the points $\mathbf{x} \in \mathcal{S}$ that are equidistant from two fixed points $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \in \text{int}(\mathcal{S})$ in terms of a Bregman divergence is a hyperplane, i.e., the partitions induced by Bregman divergences have linear separators given by $d_\phi(\mathbf{x}, \boldsymbol{\mu}_1) = d_\phi(\mathbf{x}, \boldsymbol{\mu}_2)$, which is equivalent to

$$\langle \mathbf{x}, \nabla\phi(\boldsymbol{\mu}_2) - \nabla\phi(\boldsymbol{\mu}_1) \rangle = (\phi(\boldsymbol{\mu}_1) - \phi(\boldsymbol{\mu}_2)) - (\langle \boldsymbol{\mu}_1, \nabla\phi(\boldsymbol{\mu}_1) \rangle - \langle \boldsymbol{\mu}_2, \nabla\phi(\boldsymbol{\mu}_2) \rangle)$$

6. **Dual Divergences:** Bregman divergences obtained from Legendre functions $\phi$ and their conjugates $\psi$ satisfy the following duality:

$$d_\phi(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) = \phi(\boldsymbol{\mu}_1) + \psi(\boldsymbol{\theta}_2) - \langle \boldsymbol{\mu}_1, \boldsymbol{\theta}_2 \rangle = d_\psi(\boldsymbol{\theta}_2, \boldsymbol{\theta}_1),$$

where $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \in \text{int}(\mathcal{S})$ are related to $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \text{int}(\Theta)$ by the Legendre transformation.

7. **Relation to KL-divergence** Let $\mathcal{F}_\psi$ be an exponential family with $\psi$ as the cumulant function. Then, the KL divergence between two members $p_{(\psi,\boldsymbol{\theta}_1)}$ and $p_{(\psi,\boldsymbol{\theta}_2)}$ in $F_\psi$ corresponding to natural parameters $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ can be expressed as a Bregman divergence in two possible ways. In particular,

$$KL(p_{(\psi,\theta_1)}||p_{(\psi,\theta_2)}) = d_\phi(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) = d_\psi(\boldsymbol{\theta}_2, \boldsymbol{\theta}_1)$$

where $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are the expectation parameters corresponding to $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. Further, if $\psi(\mathbf{0}) = 0$, then $p_{(\psi,\mathbf{0})}(x) = p_0(\mathbf{x})$ is itself a valid probability density and $KL(p_{(\psi,\boldsymbol{\theta})} \| \ p_{(\psi,\mathbf{0})}) = \phi(\boldsymbol{\mu})$, where $\boldsymbol{\mu} = \nabla\psi(\boldsymbol{\theta})$.

8. **Generalized Pythagoras theorem:** For any $\mathbf{x}_1 \in \mathcal{S}$ and $\mathbf{x}_2, \mathbf{x}_3 \in \text{int}(\mathcal{S})$,

$$d_\phi(\mathbf{x}_1, \mathbf{x}_2) + d_\phi(\mathbf{x}_2, \mathbf{x}_3) = d_\phi(\mathbf{x}_1, \mathbf{x}_3) + \langle \mathbf{x}_1 - \mathbf{x}_2, \nabla\phi(\mathbf{x}_3) - \nabla\phi(\mathbf{x}_2) \rangle.$$

When the points, $\mathbf{x}_1, \mathbf{x}_2$ and $\mathbf{x}_3$ are such that $\mathbf{x}_1 \in \mathcal{S}'$ where $\mathcal{S}'$ is a convex subset of $\mathcal{S}$ and $\mathbf{x}_2$ is given by

$$\mathbf{x}_2 = \underset{\mathbf{x} \in \mathcal{S}'}{\text{argmin}} \ d_\phi(\mathbf{x}, \mathbf{x}_3),$$

then the inner product term becomes negative and we have,

$$d_\phi(\mathbf{x}_1, \mathbf{x}_2) + d_\phi(\mathbf{x}_2, \mathbf{x}_3) \leq d_\phi(\mathbf{x}_1, \mathbf{x}_3).$$

When the convex subset $\mathcal{S}'$ is an affine subspace, then the inner product term becomes zero giving rise to an equality.

### Necessary and Sufficient Conditions for Bregman Divergence

A divergence measure $d : \mathcal{S} \times \text{int}(\mathcal{S}) \mapsto [0, \infty)$ is a Bregman divergence if and only if there exists $\mathbf{a} \in \text{int}(\mathcal{S})$ such that the function $\phi_\mathbf{a} = d(\mathbf{x}, \mathbf{a})$ satisfies the following

conditions.

1. $\phi_{\mathbf{a}}$ is strictly convex on $\mathcal{S}$ and differentiable on $\text{int}(\mathcal{S})$.

2. $d(\mathbf{x}, \mathbf{y}) = d_{\phi_{\mathbf{a}}}(\mathbf{x}, \mathbf{y}),\ \forall \mathbf{x} \in \mathcal{S}, \mathbf{y} \in \text{int}(\mathcal{S})$ where $d_{\phi_{\mathbf{a}}}$ is the Bregman divergence associated with $\phi_{\mathbf{a}}$.

It is easy to see the sufficiency property from the second condition. To prove that the conditions are necessary as well, we note that for any strictly convex, differentiable function $\phi$, the Bregman divergence evaluated with a fixed value for the second argument differs from it only by a linear term, i.e.,

$$\begin{aligned} \phi_{\mathbf{a}}(\mathbf{x}) = d_{\phi}(\mathbf{x}, \mathbf{a}) &= \phi(\mathbf{x}) - \phi(\mathbf{a}) - \langle \mathbf{x} - \mathbf{a}, \nabla\phi(\mathbf{a}) \rangle \\ &= \phi(\mathbf{x}) + \langle \mathbf{b}, \mathbf{x} \rangle + c, \end{aligned}$$

where $\mathbf{b} = -\nabla\phi(\mathbf{a})$ and $c = \langle \mathbf{a}, \nabla\phi(\mathbf{a}) \rangle - \phi(\mathbf{a})$. Hence, $\phi_{\mathbf{a}}$ is also strictly convex and differentiable and the Bregman divergences associated with $\phi$ and $\phi_{\mathbf{a}}$ are identical.

# Appendix B

# Exponential Family and Bregman Divergences

## B.1 Proof of Theorem 4

This section provides a proof of Theorem 4 and other related results. We begin with definitions. Let $P_0$ be any non-negative bounded measure on $\mathbb{R}^d$ and $\mathcal{F}_\psi = \{p_{(\psi,\boldsymbol{\theta})},\ \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d\}$ be a regular exponential family with dominating measure $P_0$, as discussed in section 4.1.1, and cumulant function $\psi$. For ease of exposition, we will consider $P_0$ to be a probability measure. Note that since any non-negative bounded measure can be simply converted to a probability measure by a multiplicative constant, our analysis remains practically unchanged in the general case, except for an additive constant to the cumulant function. Let $I_\psi$ be the support of $P_0$ (Definition 6) and hence, of all the probability measures in $\mathcal{F}_\psi$. Let $\phi$ be the convex conjugate of $\psi$ so that $(\text{int}(\text{dom}(\phi)), \phi)$ and $(\Theta, \psi)$ are Legendre duals of each other.

First, we focus on proving Theorem 4, i.e., $I_\psi \subseteq \text{dom}(\phi)$ using the following two lemmas (Lemmas 11, 12). Then, we present a related result connecting the

closed convex hull of $I_\psi$ and the domain of $\phi$ in a more general sense. Most of the ideas used in this analysis are from section 9.1 of [BN78]. For the sake of completeness and also since all the proofs are not provided in [BN78], we give detailed proofs of all the lemmas we use.

**Lemma 11** *For any $\boldsymbol{\theta} \in \Theta$ and $\mathbf{x} \in \mathbb{R}^d$,*

$$\langle \boldsymbol{\theta}, \mathbf{x} \rangle - \psi(\boldsymbol{\theta}) \leq -\log \left( \inf_{\mathbf{e}} P_0 \left[ \langle \mathbf{e}, X \rangle \geq \langle \mathbf{e}, \mathbf{x} \rangle \right] \right)$$

*where $X \sim P_0$ and the infimum is taken over all the unit vectors $\mathbf{e}$ in $\mathbb{R}^d$.*

*Proof:* Let $\mathbf{e}_{\boldsymbol{\theta}}$ be the unit vector in the direction of $\boldsymbol{\theta}$. Given any $\mathbf{x} \in \mathbb{R}^d$, it is possible to divide $\mathbb{R}^d$ into two half spaces $\mathcal{G}_1 = \{\mathbf{x}' \in \mathbb{R}^d | \ \langle \mathbf{e}_{\boldsymbol{\theta}}, \mathbf{x}' \rangle < \langle \mathbf{e}_{\boldsymbol{\theta}}, \mathbf{x} \rangle\}$ and $\mathcal{G}_2 = \{\mathbf{x}' \in \mathbb{R}^d | \ \langle \mathbf{e}_{\boldsymbol{\theta}}, \mathbf{x}' \rangle \geq \langle \mathbf{e}_{\boldsymbol{\theta}}, \mathbf{x} \rangle\}$. Now, for any $\boldsymbol{\theta}$, we have

$$\begin{aligned} & & 1 &= \int_{\mathbf{x}' \in \mathbb{R}^d} \exp(\langle \boldsymbol{\theta}, \mathbf{x}' \rangle - \psi(\boldsymbol{\theta})) dP_0(\mathbf{x}') \\ \Rightarrow & & \exp(\psi(\boldsymbol{\theta})) &= \int_{\mathbf{x}' \in \mathbb{R}^d} \exp(\langle \boldsymbol{\theta}, \mathbf{x}' \rangle) dP_0(\mathbf{x}') \ . \end{aligned}$$

Partitioning the integral over $\mathbb{R}^d$ into $\mathcal{G}_1$ and $\mathcal{G}_2$, we obtain

$$\begin{aligned} \exp(\psi(\boldsymbol{\theta})) &= \int_{\mathbf{x}' \in \mathcal{G}_1} \exp(\langle \boldsymbol{\theta}, \mathbf{x}' \rangle) dP_0(\mathbf{x}') + \int_{\mathbf{x}' \in \mathcal{G}_2} \exp(\langle \boldsymbol{\theta}, \mathbf{x}' \rangle) dP_0(\mathbf{x}') \\ &\geq \int_{\mathbf{x}' \in \mathcal{G}_2} \exp(\langle \boldsymbol{\theta}, \mathbf{x}' \rangle) dP_0(\mathbf{x}') \\ &\geq \exp(\langle \boldsymbol{\theta}, \mathbf{x} \rangle) \int_{\mathbf{x}' \in \mathcal{G}_2} dP_0(\mathbf{x}') \quad (\because \langle \mathbf{e}_{\boldsymbol{\theta}}, \mathbf{x}' \rangle \geq \langle \mathbf{e}_{\boldsymbol{\theta}}, \mathbf{x} \rangle \text{ for } \mathbf{x}' \in \mathcal{G}_2 \ ) \\ &= \exp(\langle \boldsymbol{\theta}, \mathbf{x} \rangle) P_0[\langle \mathbf{e}_{\boldsymbol{\theta}}, X \rangle \geq \langle \mathbf{e}_{\boldsymbol{\theta}}, \mathbf{x} \rangle] \\ &\geq \exp(\langle \boldsymbol{\theta}, \mathbf{x} \rangle) \inf_{\mathbf{e}} P_0[\langle \mathbf{e}, X \rangle \geq \langle \mathbf{e}, \mathbf{x} \rangle] \ . \end{aligned}$$

On taking logarithms and re-arranging terms, we obtain the desired result. ∎

**Lemma 12** *For any* $\mathbf{x} \in \mathbb{R}^d$,

$$\inf_{\mathbf{e}} P_0[\langle \mathbf{e}, X \rangle \geq \langle \mathbf{e}, \mathbf{x} \rangle] > 0 \quad \Rightarrow \quad \mathbf{x} \in \mathrm{dom}(\phi)$$

*where* $X \sim P_0$ *and the infimum is taken over all the unit vectors* $\mathbf{e}$ *in* $\mathbb{R}^d$.

*Proof:* Let $\rho(\mathbf{x}) = \inf_{\mathbf{e}} P_0[\langle \mathbf{e}, X \rangle \geq \langle \mathbf{e}, \mathbf{x} \rangle] > 0$. From Lemma 11, we know that

$$\langle \boldsymbol{\theta}, \mathbf{x} \rangle - \psi(\boldsymbol{\theta}) \leq -\log(\rho(\mathbf{x})), \ \forall \mathbf{x} \in \mathbb{R}^d.$$

Hence, $\rho(\mathbf{x}) > 0$ implies that $\forall \boldsymbol{\theta}, \langle \boldsymbol{\theta}, \mathbf{x} \rangle - \psi(\boldsymbol{\theta}) \leq -\log(\rho(\mathbf{x})) < \infty$, so that

$$\phi(\mathbf{x}) = \sup_{\boldsymbol{\theta}} (\langle \boldsymbol{\theta}, \mathbf{x} \rangle - \psi(\boldsymbol{\theta})) \leq -\log(\rho(\mathbf{x})) < \infty,$$

i.e., $\mathbf{x} \in \mathrm{dom}(\phi)$. ∎

**Proof of Theorem 4** Let $\mathbf{x} \in I_\psi$ and let $\mathbf{e}$ be any unit vector. Let $H(\mathbf{e}, \mathbf{x})$ be the hyperplane through $\mathbf{x}$ with unit normal $\mathbf{e}$. Let $\mathcal{H}(\mathbf{e}, \mathbf{x})$ be the closed half-space determined by the hyperplane $H(\mathbf{e}, \mathbf{x})$ and the unit vector $\mathbf{e}$, i.e., the set $\mathcal{H}(\mathbf{e}, \mathbf{x}) = \{\mathbf{x} \in \mathbb{R}^d | \langle \mathbf{e}, X \rangle \geq \langle \mathbf{e}, \mathbf{x} \rangle\}$. Using this notation, we give separate proofs for the cases when $P_0$ is absolutely continuous with respect to the counting measure and with respect to the Lebesgue measure.

Let $P_0$ be absolutely continuous with respect to the counting measure. Now, by definition, $\mathcal{H}(\mathbf{e}, \mathbf{x})$ always contains $\mathbf{x}$. Since $\mathbf{x} \in I_\psi$, applying Definition 6 to the set $S = \{\mathbf{x}\}$ we have $p_{(\psi, \boldsymbol{\theta})}(\mathbf{x}) > 0$. Hence $p_0(\mathbf{x}) > 0$ as the exponential family distribution is absolutely continuous with respect to $P_0$. Therefore, the closed half-

space $\mathcal{H}(\mathbf{e}, \mathbf{x})$ has a positive measure of at least $p_0(\mathbf{x})$ for any unit vector $\mathbf{e}$, i.e.,

$$\begin{aligned} & P_0\{\langle \mathbf{e}, X\rangle \geq \langle \mathbf{e}, \mathbf{x}\rangle\} \;\; \geq \;\; p_0(\mathbf{x}) \;\; > \;\; 0 \quad \forall \mathbf{e} \\ \Rightarrow \qquad & \inf_{\mathbf{e}} P_0\{\langle \mathbf{e}, X\rangle \geq \langle \mathbf{e}, \mathbf{x}\rangle\} \;\; \geq \;\; p_0(\mathbf{x}) \;\; > \;\; 0 \; . \end{aligned}$$

From Lemma 12, it follows that $\mathbf{x} \in \mathrm{dom}(\phi)$. Therefore, $I_\psi \subseteq \mathrm{dom}(\phi)$.

Now, we consider the case when $P_0$ is absolutely continuous with respect to the Lebesgue measure. If $\mathbf{x} \in I_\psi$, then $\forall S \subseteq \mathbb{R}^d$ with $\mathbf{x} \in S$ and $\int_S d\mathbf{x} > 0$, we have

$$\int_S p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}) d\mathbf{x} > 0 \; .$$

Note that since $\mathbf{x} \in \mathcal{H}(\mathbf{e}, \mathbf{x})$, $\int_{\mathcal{H}(\mathbf{e},\mathbf{x})} d\mathbf{x} > 0$ and further $\mathbf{x} \in I_\psi$, we must have

$$\int_{\mathcal{H}(\mathbf{e},\mathbf{x})} p_{(\psi,\boldsymbol{\theta})}(\mathbf{x}) d\mathbf{x} > 0 \quad \forall \mathbf{e}.$$

Since the exponential family distribution is absolutely continuous with respect to $P_0$, we have $P_0(\mathcal{H}(\mathbf{e}, \mathbf{x})) > 0, \forall \mathbf{e} \Rightarrow P_0(\langle \mathbf{e}, X\rangle \geq \langle \mathbf{e}, \mathbf{x}\rangle) > 0, \forall \mathbf{e}$. Now, since the set of unit vectors is a compact set, $\inf_{\mathbf{e}} \; P_0(\langle \mathbf{e}, X\rangle \geq \langle \mathbf{e}, \mathbf{x}\rangle)$ is achieved by some $\mathbf{e}^*$ in the set of unit vectors itself, so that

$$\inf_{\mathbf{e}} \; P_0(\langle \mathbf{e}, X\rangle \geq \langle \mathbf{e}, \mathbf{x}\rangle) = P_0(\langle \mathbf{e}^*, X\rangle \geq \langle \mathbf{e}^*, \mathbf{x}\rangle) > 0 \; .$$

Again, from Lemma 12 it follows that $\mathbf{x} \in \mathrm{dom}(\phi)$ so that $I_\psi \subseteq \mathrm{dom}(\phi)$. ∎

## B.2 A Related Theorem

We now present a more general theorem involving the closed convex hull of $I_\psi$. The result is not essential to chapter 4, but may be interesting in its own right.

**Theorem 19** *Let $I_\psi$ be as in Definition 6. Let $C_\psi$ be the closure of the convex hull of $I_\psi$, i.e., $C_\psi = \mathrm{co}(I_\psi)$. Then,*

$$\mathrm{int}(C_\psi) \subseteq \mathrm{dom}(\phi) \subseteq C_\psi$$

*where $\phi$ is the convex conjugate of $\psi$.*

In order to prove the above theorem, we need a few additional lemmas.

**Lemma 13** *Let $C_\psi = \mathrm{co}(I_\psi)$. Then,* $\mathrm{int}(C_\psi) \subseteq \mathrm{dom}(\phi)$.

The proof of this result requires the following result from [Val64].

**Lemma 14** *For any set $\mathcal{G} \subseteq \mathbb{R}^d$, if $\mathbf{x} \in \mathrm{int}(\mathrm{co}(\mathcal{G}))$, then there exists a positive integer $m \leq 2d$ and points $\mathbf{x}_1, \cdots, \mathbf{x}_m \in \mathcal{G}$ such that $\mathbf{x} \in \mathrm{int}(\mathrm{co}(\mathbf{x}_1, \cdots, \mathbf{x}_m))$.*

**Proof of Lemma 13** Let $\mathbf{x} \in \mathrm{int}(C_\psi)$. Using the fact that $C_\psi = \mathrm{co}(I_\psi)$ and Lemma 14, we observe that $\mathbf{x} \in \mathrm{int}(\mathrm{co}(\mathbf{a}_1, \cdots, \mathbf{a}_m))$ where $\mathbf{a}_1, \cdots, \mathbf{a}_m \in I_\psi$ and $m \leq 2d$. Let $H(\mathbf{e}, \mathbf{x})$ be the hyperplane through $\mathbf{x}$ with unit normal $\mathbf{e}$ and let $\delta_i(\mathbf{e})$ be the distance from $\mathbf{a}_i$ to the hyperplane $H(\mathbf{e}, \mathbf{x})$ and let $\delta(\mathbf{e}) = \max\limits_{1 \leq i \leq m} \delta_i(\mathbf{e})$. The mapping $\delta$ defined on the set of unit vectors is a continuous mapping on a compact set and hence, it attains an infimum $\delta_0$. Clearly $\delta_0 > 0$ since $\delta_0 = 0$ would imply that $\mathbf{a}_i \in H(\mathbf{e}, \mathbf{x})$ for some unit vector $\mathbf{e}$, which violates $\mathbf{x} \in \mathrm{int}(\mathrm{co}(\mathbf{a}_1, \cdots, \mathbf{a}_m))$. Consequently, for every unit vector $\mathbf{e}$, the closed positive space determined by $H(\mathbf{e}, \mathbf{x})$ and $\mathbf{e}$ contains at least one of the open balls $B(\mathbf{a}_i, \delta_0)$ with center $\mathbf{a}_i$, $1 \leq i \leq m$ and radius $\delta_0$ and each of these balls have positive $P_0$ measure since $\mathbf{a}_i \in I_\psi$. Hence,

$$\inf_{\mathbf{e}} P_0\{\langle \mathbf{e}, X \rangle \geq \langle \mathbf{e}, \mathbf{x} \rangle\} \geq \min_{1 \leq i \leq m} P_0(B(\mathbf{a}_i, \delta_0)) > 0 \ .$$

Now, from Lemma 12, it follows that $\mathbf{x} \in \mathrm{dom}(\phi)$. Therefore, $\mathrm{int}(C_\psi) \subseteq \mathrm{dom}(\phi)$.

■

**Lemma 15** *Let $C_\psi = \mathrm{co}(I_\psi)$. Then,* $\mathrm{dom}(\phi) \subseteq C_\psi$.

*Proof:* Let $\mathbf{x} \notin C_\psi$. Since $C_\psi$ is a closed set, there exists a hyperplane $H$ that strongly separates $\mathbf{x}$ and $C_\psi$. Let $\mathbf{e}$ be the unit vector in $\mathbb{R}^d$ that is normal to $H$ such that $C_\psi$ lies in the negative half space determined by $H$ and $\mathbf{e}$. Then, $\mathbf{x}$ lies in the positive half space. Therefore, for all $\mathbf{a} \in C_\psi$, we have $\langle \mathbf{e}, \mathbf{x} \rangle \geq \langle \mathbf{e}, \mathbf{a} \rangle + \delta$ for some $\delta > 0$. For any $\alpha > 0$, let $\boldsymbol{\theta}_\alpha = \alpha \mathbf{e}$ so that $\alpha = \|\boldsymbol{\theta}_\alpha\|$. Then, $\forall \mathbf{a} \in C_\psi, \langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle \geq \langle \boldsymbol{\theta}_\alpha, \mathbf{a} \rangle + \alpha\delta$. Then, by definition,

$$\begin{aligned}
\exp(\psi(\boldsymbol{\theta}_\alpha)) &= \int_{\mathbf{x}' \in \mathbb{R}^d} \exp(\langle \boldsymbol{\theta}_\alpha, \mathbf{x}' \rangle)\, dP_0(\mathbf{x}') \\
&= \int_{\mathbf{x}' \in I_\psi \subseteq C_\psi} \exp(\langle \boldsymbol{\theta}_\alpha, \mathbf{x}' \rangle)\, dP_0(\mathbf{x}') \\
&= \exp(\langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle) \int_{\mathbf{x}' \in I_\psi \subseteq C_\psi} \exp(\langle \boldsymbol{\theta}_\alpha, \mathbf{x}' \rangle - \langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle)\, dP_0(\mathbf{x}') \\
&\leq \exp(\langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle) \int_{\mathbf{x}' \in I_\psi \subseteq C_\psi} \exp(-\alpha\delta)\, dP_0(\mathbf{x}') \\
&= \exp(\langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle - \alpha\delta) \qquad (\because \textstyle\int_{\mathbf{x}' \in I_\psi \subseteq C_\psi} dP_0(\mathbf{x}') = 1 ) \ .
\end{aligned}$$

By taking logarithms and rearranging terms, we obtain

$$\begin{aligned}
\langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle - \psi(\boldsymbol{\theta}_\alpha) &\geq \alpha\delta \\
\Rightarrow \qquad \lim_{\alpha \to \infty} (\langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle - \psi(\boldsymbol{\theta}_\alpha)) &\geq \lim_{\alpha \to \infty} \alpha\delta \ = \ \infty \ .
\end{aligned}$$

Now,

$$\begin{aligned}
\phi(\mathbf{x}) = \sup_{\boldsymbol{\theta} \in \mathbb{R}^d} (\langle \boldsymbol{\theta}, \mathbf{x} \rangle - \psi(\boldsymbol{\theta})) &\geq \langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle - \psi(\boldsymbol{\theta}_\alpha), \ \forall \alpha \\
\Rightarrow \qquad \phi(\mathbf{x}) &\geq \lim_{\alpha \to \infty} (\langle \boldsymbol{\theta}_\alpha, \mathbf{x} \rangle - \psi(\boldsymbol{\theta}_\alpha)) \ \geq \ \lim_{\alpha \to \infty} \alpha\delta \\
\Rightarrow \qquad \phi(\mathbf{x}) &= \infty \ .
\end{aligned}$$

Hence, $\mathbf{x} \notin \text{dom}(\phi)$. Therefore, $\text{dom}(\phi) \subseteq C_\psi$. ■

Theorem 19 follows directly from Lemmas 13 and 15. In addition to this theorem, there are other results that specify the relation between $C_\psi$ and $\text{dom}(\phi)$ more exactly for special cases. In particular, when $I_\psi$ is finite or countable, it can be shown [BN78, Theorem 9.4] that $\text{dom}(\phi) = C_\psi$. Further, it can also be proved [BN78, Theorem 9.5] that if $\mathbf{x} \in \text{bd}(C_\psi)$ and $\mathbf{x} \notin I_\psi$, then $\mathbf{x} \notin \text{dom}(\phi)$.

# Appendix C

# Rate Distortion Theory for Bregman Divergences

## C.1 Proof of Theorem 9

To prove the theorem on Shannon-Bregman lower bound, we need the following well-known theorem on the rate distortion function for general distortion measures.

**Theorem 20 ([Ber71])** *Let $X$ be a source with a distribution $p(x)$ over $x$, $\hat{\mathcal{X}}$ be the reproduction alphabet and $d(\cdot,\cdot)$ the distortion measure. Let $\Lambda_\beta$ be the set of all non-negative functions $\lambda(x)$ satisfying*

$$c(\hat{x}) = \int_x \lambda(x)p(x)\exp(-\beta d(x,\hat{x}))dx \leq 1 \ \forall \hat{x} \tag{C.1}$$

*Then, the rate distortion function $R(D)$ is given by*

$$R(D) = \sup_{\beta\geq 0, \lambda\in\Lambda_\beta} \{-\beta D + \int_x p(x)\log\lambda(x)dx\}. \tag{C.2}$$

*Further, for each $\beta \geq 0$, a necessary and sufficient condition for $\lambda(x)$ to realize the*

*supremum in (C.2) is the existence of a probability density $q(\hat{x})$ that is related to $\lambda(x)$ by the following relation*

$$\lambda(x) = \left( \int_x q(\hat{x}) \exp(-\beta d(x, \hat{x})) d\hat{x} \right)^{-1} \tag{C.3}$$

*and is such that $c(\hat{x}) = 1$ for almost all $\hat{x}$ for which $q(\hat{x}) \geq 0$.*

*Proof of theorem 9 :* From the bijection theorem (Theorem 5) and the linearity of Bregman divergences [BMDG03], we note that for every Bregman divergence $d_\phi$ and a choice of $\beta \geq 0$, there exists a unique non-negative function $f_{\beta\phi}(x)$ such that

$$\int_x \exp(-d_{\beta\phi}(x, \hat{x})) f_{\beta\phi}(x) dx = 1$$

for all $\hat{x}$. Defining $\lambda_{(L,\beta)}(x) = \frac{f_{\beta\phi}(x)}{p(x)};\ p(x) \neq 0$, we observe that the function

$$c_L(\hat{x}) = \int_x \lambda_{(L,\beta)}(x) p(x) \exp(-\beta d(x, \hat{x})) dx = \int_x f_{\beta\phi} \exp(-\beta d(x, \hat{x})) dx = 1,$$

so that $\lambda_{(L,\beta)} \in \Lambda_\beta$, i.e., the set of eligible partition functions mentioned in theorem 20. Therefore,

$$\begin{aligned}
R(D) &= \sup_{\beta \geq 0, \lambda \in \Lambda_\beta} \{-\beta D + \int_x p(x) \log \lambda(x) dx\} \\
&\geq \sup_{\beta \geq 0} \{-\beta D + \int_x p(x) \log \lambda_{(L,\beta)}(x) dx\} \\
&\geq \sup_{\beta \geq 0} \{-\beta D + \int_x p(x) \log \frac{f_{\beta\phi}(x)}{p(x)} dx\} \\
&\geq h(p(x)) + \sup_{\beta \geq 0} \{-\beta D + \int_x p(x) \log f_{\beta\phi}(x) dx\} = R_L(D).
\end{aligned}$$

That completes the proof. ∎

## C.2 Proof of Theorem 10

First, we provide the proof of a slightly weaker result. We prove that the existence of a non-empty open ball in the support of the optimal reproduction random variable for a given distortion implies that the Shannon-Bregman lower bound for the rate distortion holds with equality at that distortion.

*Proof of weaker version of theorem 9 :* The rate distortion problem involves optimizing the objective function $J_1(p(\hat{x}|x)) = I(X;\hat{X}) + \beta \int_{x,\hat{x}} p(x)p(\hat{x}|x)d_\phi(x,\hat{x})dx$, where $\beta$ is the variational parameter corresponding to the desired distortion level. Rewriting the objective function $J_1(p(\hat{x}|x))$ in terms of $p(\hat{x})$ using the stationary point conditions (corollary 8), we obtain

$$\begin{aligned}
J_1(p(\hat{x}|x)) &= I(X;\hat{X}) + \beta \int_{x,\hat{x}} p(x)p(\hat{x}|x)d_\phi(x,\hat{x})dxd\hat{x} \\
&= \int_{x,\hat{x}} p(x)p(\hat{x}|x)\log\frac{p(\hat{x}|x)}{p(\hat{x})}dxd\hat{x} + \beta\int_{x,\hat{x}} p(x)p(\hat{x}|x)d_\phi(x,\hat{x})dxd\hat{x} \\
&= \int_{x,\hat{x}} p(x)p(\hat{x}|x)\log\frac{p(\hat{x}|x)}{p(\hat{x})}dxd\hat{x} + \beta\int_{x,\hat{x}} p(x)p(\hat{x}|x)d_\phi(x,\hat{x})dxd\hat{x} \\
&\overset{(a)}{=} \int_{x,\hat{x}} p(x)p(\hat{x}|x)(-\beta d_\phi(x,\hat{x}) - \log N(x,\beta))dxd\hat{x} \\
&\qquad\qquad + \beta\int_{x,\hat{x}} p(x)p(\hat{x}|x)d_\phi(x,\hat{x})dxd\hat{x} \\
&= -\int_{x,\hat{x}} p(x)p(\hat{x}|x)\log N(x,\beta)dxd\hat{x} = -\int_x p(x)\log N(x,\beta)dx \\
&= -\int_x p(x)\log\left(\int_{\hat{x}} p(\hat{x})\exp(-\beta d_\phi(x,\hat{x}))d\hat{x}\right)dx = J_2(p(\hat{x}))\ ,
\end{aligned}$$

where (a) follows since $p(\hat{x}|x) = \frac{p(\hat{x})}{N(x,\beta)}\exp(-\beta d_\phi(x,\hat{x}))$ where the normalization term $N(x,\beta) = \int_{\hat{x}} p(\hat{x})\exp(-\beta d_\phi(x,\hat{x}))d\hat{x}$. Solving the rate distortion problem now involves finding the optimal distribution $p(\hat{x})$ over the reproduction alphabet $\hat{\mathcal{X}}$. An alternate way to address this problem is through the mapping approach [Ros94], where instead of searching for the optimal $p(\hat{x}$, we search for the optimal map-

ping from the unit interval to the reproduction alphabet such that the Lebesgue measure over the unit interval results in the optimal $p(\hat{x})$ over the reproduction alphabet. The equivalence of the two approaches is guaranteed by the Borel isomorphism theorem [KF75], which states that a complete separable metric space with a finite Borel measure (in this case $\hat{\mathcal{X}}$ with $p(\hat{x})$) is isomorphic to the unit interval with the Lebesgue measure (denoted by $\mu$). Hence, for each probability measure corresponding to $p(\hat{x})$ on $\hat{\mathcal{X}}$, there exists a unique mapping $\hat{x} : [0,1] \mapsto \hat{\mathcal{X}}$ that maps the Lebesgue measure to $p(\hat{x})$ such that for any function $f$ defined over $\hat{x}$, $\int_{\hat{x}} f(\hat{x})p(\hat{x})d\hat{x} = \int_{u\in[0,1]} f(\hat{x}(u))d\mu(u)$. Therefore, the rate distortion objective function can now be rewritten as

$$\begin{aligned} J_2(p(\hat{x})) &= -\int_x p(x)\log\left(\int_{\hat{x}} p(\hat{x})\exp(-\beta d_\phi(x,\hat{x}))d\hat{x}\right)dx \\ &= -\int_x p(x)\log\left(\int_{u\in[0,1]} \exp(-\beta d_\phi(x,\hat{x}(u)))d\mu u\right)dx = J_3(\hat{x}) \end{aligned}$$

and the necessary condition for optimality is obtained using variational calculus as

$$-\int_x p(x)\frac{1}{\int_{u\in[0,1]}\exp(-\beta d_\phi(x,\hat{x}(u)))d\mu u}\left(\frac{d}{d\hat{x}}\exp(-\beta d_\phi(x,\hat{x}(u)))\right)dx = 0.$$

Let $\hat{\mathcal{X}}_s$ be the support of the optimal reproduction random variable, i.e., the subset of $\hat{\mathcal{X}}$ for which $p(\hat{x} \neq 0$. From the isomorphism result, $\hat{\mathcal{X}}_s$ is also the range of the optimal mapping $\hat{x}(u)$ and so every point $\hat{x} \in \hat{\mathcal{X}}_s$ satisfies the optimality condition mentioned above, i.e.,

$$-\int_x p(x)\frac{1}{\int_{\hat{x}}\exp(-\beta d_\phi(x,\hat{x}))p(\hat{x})d\hat{x}}\left(\frac{d}{d\hat{x}}\exp(-\beta d_\phi(x,\hat{x}))\right)dx = 0. \qquad \text{(C.4)}$$

Let us now assume that the optimal support $\hat{\mathcal{X}}_s$ contains a non-empty open ball $B_\epsilon$, $\epsilon > 0$. Then, it follows that the optimality condition (C.4) holds for all $\hat{x} \in B_\epsilon$

and hence,

$$\int_x p(x) \frac{1}{\int_{\hat{x}} \exp(-\beta d_\phi(x,\hat{x}))p(\hat{x})d\hat{x}} \exp(-\beta d_\phi(x,\hat{x}))dx$$
$$= \int_x \frac{p(x)}{N(x,\beta)} \exp(-\beta d_\phi(x,\hat{x}))dx = k_0 \ .$$

for all $\hat{x} \in B_\epsilon \subseteq \hat{\mathcal{X}}_s$ where $k_0$ is a constant. Considering the expectation of the left hand side expression over the open ball $B_\epsilon$ gives us

$$\begin{aligned}
\int_{\hat{x}\in B_\epsilon} \int_x \frac{p(x)p(\hat{x})}{N(x,\beta)} \exp(-\beta d_\phi(x,\hat{x}))dxd\hat{x} &= k_0 \int_{\hat{x}\in B_\epsilon} p(\hat{x})d\hat{x} \\
\Rightarrow \int_{\hat{x}\in B_\epsilon} \int_x p(x)p(\hat{x}|x)dxd\hat{x} &= k_0 \int_{\hat{x}\in B_\epsilon} p(\hat{x})d\hat{x} \\
\Rightarrow \int_{\hat{x}\in B_\epsilon} p(\hat{x}) \int_x p(x|\hat{x})dxd\hat{x} &= k_0 \int_{\hat{x}\in B_\epsilon} p(\hat{x})d\hat{x} \\
\Rightarrow 1 &= k_0
\end{aligned}$$

Now, setting $\lambda^*_\beta(x) = \frac{1}{N(x,\beta)} = (\int_{\hat{x}} p(\hat{x}) \exp(-\beta d_\phi(x,\hat{x}))d\hat{x})^{-1}$, we obtain

$$\int_x p(x)\lambda^*_\beta(x) \exp(-\beta d_\phi(x,\hat{x}))dx = 1 \ \forall \hat{x} \in B_\epsilon \tag{C.5}$$

From the bijection theorem (Theorem 5), this implies that $p(x)\lambda^*_\beta(x) = f_{\beta\phi}(x), \ \forall x \in x$ so that the expression

$$c^*(\hat{x}) = \int_x \lambda^*_\beta(x)p(x) \exp(-\beta d(x,\hat{x}))dx = \int_x f_{\beta\phi}(x) \exp(-\beta d(x,\hat{x}))dx = 1 \ .$$

Therefore, the function $\lambda^*_\beta(x)$ satisfies the conditions (C.1) and (C.3) mentioned in theorem 20 and attains the supremum in the (C.2) for the given $\beta$, i.e.

$$R(D_\beta) = -\beta D_\beta + \int_x p(x) \log \lambda^*_\beta(x)dx$$

where $D_\beta$ is the distortion value for which the supremum is attained at the given $\beta$. Further, $\lambda^*_\beta(x) = \frac{f_{\beta\phi}(x)}{p(x)} = \lambda_{(L,\beta)}(x), \forall x \in x,\ p(x) \neq 0$ and so, the Shannon-Bregman lower bound

$$R_L(D_\beta) = -\beta D_\beta + \int_x p(x) \log \lambda^*_\beta(x) dx = R(D_\beta).$$

That completes the proof. ■

*Proof of theorem 9 :* From the condition of the theorem, the optimal support $\hat{\mathcal{X}}_s$ contains an accumulation point $\hat{x}_0$, i.e., for every $\epsilon > 0$, there exists a $\sigma$ such that $0 < |\sigma| < \epsilon$ and $\hat{x}_0 + \sigma$ is a point in the optimal support $\hat{\mathcal{X}}_s$ of $Z$. Then, following the previous argument, all points in the support must satisfy the optimality equation (see (C.4))

$$\int_x p(x)\lambda^*_\beta(x) \left( \frac{d}{d\hat{x}} \exp(-\beta d_\phi(x, \hat{x}_0 + \sigma) \right) dx = 0 \qquad \forall\ \hat{x}_0 + \sigma \in \hat{\mathcal{X}}_s \tag{C.6}$$

where

$$\lambda^*_\beta(x) = \frac{1}{N(x,\beta)} = \left[ \int_{\hat{x}} p(\hat{x}) \exp(-\beta d_\phi(x, \hat{x})) d\hat{x} \right]^{-1} .$$

The condition can be rewritten as

$$\frac{d}{d\hat{x}} \int_x p(x)\lambda^*_\beta(x) \exp(-\beta d_\phi(x, \hat{x}_0 + \sigma) dx = 0 \ , \tag{C.7}$$

where the interchange is justified since the integrals in (C.6) and (C.7) are universally convergent. Now, if $(\psi, \theta_0 + \delta)$ is the conjugate of $(\phi, \hat{x}_0 + \sigma)$, then (C.7) can be equivalently written as

$$\frac{\partial}{\partial \delta} \int_x p(x)\lambda^*_\beta(x) \exp(-\beta\phi(x)) \exp(\beta((\theta_0 + \delta)x - \psi(\theta_0 + \delta)) dx = 0 \qquad \forall\ \theta_0 + \delta \in \hat{\Theta}_s, \tag{C.8}$$

where

$$\Theta_s = \{\theta_0 + \delta | \nabla\psi(\theta_0 + \delta) \in \hat{\mathcal{X}}_s\} \ ,$$

the conjugate of $\hat{\mathcal{X}}_s$. Now, note that

$$\exp(\beta((\theta_0 + \delta) - \psi(\theta_0 + \delta)) = \sum_{n=0}^{\infty} \frac{\delta^n}{n!} \ \frac{\partial^{(n)} \exp(\beta(\theta_0 x - \psi(\theta_0))}{\partial\theta^{(n)}}$$

Replacing back in (C.8) and re-arranging summation and integration, we have

$$\sum_{n=1}^{\infty} \frac{\delta^{(n-1)}}{n!} \ \int_x p(x)\lambda_\beta^*(x) \exp(-\beta\phi(x)) \frac{\partial^{(n)} \exp(\beta(\theta_0 x - \psi(\theta_0))}{\partial\theta^{(n)}} dx = 0, \qquad \text{(C.9)}$$

for all $\delta$ such that $\theta_0 + \delta \in \hat{\Theta}_s$. Again, the rearrangement is justified by the use of Lebesgue's dominated convergence theorem [Ros94, CL00]. Since the power series over different $\delta$ form a Vandermonde matrix [CL00], which is non-singular, we have

$$\int_x p(x)\lambda_\beta^*(x) \exp(-\beta\phi(x)) \frac{\partial^{(n)} \exp(\beta(\theta_0 x - \psi(\theta_0))}{\partial\theta^{(n)}} dx = 0, \qquad n \geq 1 \ . \qquad \text{(C.10)}$$

Note that the above equation determines $\frac{\partial^{(n)}\psi(\theta_0)}{\partial\theta^{(n)}}$, for each value $n$. Since $d_\phi$ is a regular Bregman divergence, $\psi()$ is the cumulant function of a regular exponential family with a base measure, say $p_\beta(x)$. Then, with $p(x)\lambda_\beta^*(x) \exp(-\beta\phi(x)) = p_\beta(x)$, an obvious solution to the above set of equations is given by the regular exponential family distribution

$$p_{(\psi,\theta_0)} = \exp(\beta(\theta_0 x - \psi(\theta_0)))p_\beta(x) \ . \qquad \text{(C.11)}$$

It remains to show that the above distribution is the (substantially) unique distribution.

Since $d_\phi$ is a regular Bregman divergence, then, by definition, $\psi$ is the cumulant function of a regular exponential family. More generally, $\beta\psi$ gives the cumulant of the scaled exponential family [MN89]. Since the family is regular,

$\exp(\beta\psi(\theta)) < \infty$ for an open ball including 0. Hence, there exists $\delta > 0$ such that $\exp(\beta\psi(\delta)) < \infty$ as well as $\exp(\beta\psi(-\delta)) < \infty$. Then,

$$\begin{aligned}\int_{-\infty}^{\infty} p_\beta(x)\exp(\delta|x|)dx &= \int_0^{\infty} p_\beta(x)\exp(\delta x)dx + \int_{-\infty}^{0} p_\beta(x)\exp(-\delta x)dx \\ &\leq \int_{-\infty}^{\infty} p_\beta(x)\exp(\delta x)dx + \int_{-\infty}^{\infty} p_\beta(x)\exp(-\delta x)dx \\ &= \exp(\beta\psi(\delta)) + \exp(\beta\psi(-\delta)) \ < \ \infty \ .\end{aligned}$$

Then, from [Har17] and [ST43, Corollary 1.2], it follows that (C.11) is the unique distribution that satisfies the set of equations (C.10).

Hence, $p(x)\lambda_\beta^*(x)\exp(\beta\phi(x)) = p_\beta(x),\ \forall x \in \mathcal{X}$ so that the expression

$$c^*(\hat{x}) = \int_x \lambda_\beta^*(x)p(x)\exp(-\beta d(x,\hat{x}))dx = \int_x \exp(-\beta(\theta_0 x - \psi(\theta_0)))p_\beta(x)dx = 1.$$

Therefore, the function $\lambda_\beta^*(x)$ satisfies the conditions (C.1) and (C.3) mentioned in theorem 20 and attains the supremum in (C.2) for the given $\beta$, i.e.

$$R(D_\beta) = -\beta D_\beta + \int_x p(x)\log\lambda_\beta^*(x)dx,$$

where $D_\beta$ is the distortion value for which the supremum is attained at the given $\beta$. Further, $\lambda_\beta^*(x) = \frac{p_\beta(x)}{p(x)} = \lambda_{(L,\beta)}(x), \forall x \in \mathcal{X},\ p(x) \neq 0$. Hence, the Shannon-Bregman lower bound

$$R_L(D_\beta) = -\beta D_\beta + \int_x p(x)\log\lambda_\beta^*(x)dx = R(D_\beta)\ .$$

That completes the proof. ∎

# Bibliography


[ABKS99] M. Ankerst, M. Breunig, H. P. Kreigel, and J. Sander. Optics: Ordering points to identify cluster structure. In *Proc. ACM SIGMOD Conf. on Management of Data*, pages 49–60, 1999.

[AGGR98] R. Agarwal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. 1998 ACM SIGMOD Intl. Conf. Management of Data*, pages 49–60, 1998.

[AKCM90] S. C. Ahalt, A. K. Krishnamurthy, P. Chen, and D. E. Melton. Competitive learning algorithms for vector quantization. *Neural Networks*, 3(3):277–290, 1990.

[Akh65] N. I. Akhizer. *The Classical Moment Problem and some related questions in analysis.* Hafner Publishing Company, 1965.

[Ama95] S. I. Amari. Information geometry of the EM and em algorithms for neural networks. *Neural Networks*, 8(9):1379–1408, 1995.

[AN01] S. Amari and H. Nagaoka. *Methods of Information Geometry.* American Mathematical Society, 2001.

[Ari72] S. Arimoto. An algorithm for computing the capacity of arbitrary dis-

crete memoryless channels. *IEEE Transactions on Information Theory*, 18:14–20, 1972.

[Ath99] K. B. Athreya. Prediction under convex loss. Technical Report 99-2, Department of Mathematics and Statistics, Iowa State University, 1999.

[AW01] K. S. Azoury and M. K. Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43(3):211–246, 2001.

[BBD00] P. S. Bradley, K. P. Bennett, and A. Demiriz. Constrained k-means clustering. Technical report, Microsoft Research, May 2000.

[BBM02] S. Basu, A. Banerjee, and R. Mooney. Semi-supervised clustering by seeding. In *Proceedings International Conference on Machine Learning*, 2002.

[BCR84] C. Berg, J. Christensen, and P. Ressel. *Harmonic Analysis on Semigroups: Theory of Positive Definite and Related Functions*. Springer-Verlag, 1984.

[BDGM04] A. Banerjee, I. Dhillon, J. Ghosh, and S. Merugu. An information theoretic analysis of maximum likelihood mixture estimation for exponential families. In *Proc. 21st Intl. Conf. on Machine Learning (ICML)*, 2004.

[BDGS03] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra. Generative model-based clustering of directional data. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 19–28, 2003.

[Ber71] T. Berger. *Rate Distortion Theory: A Mathematical Basis for Data Compression*. Prentice-Hall, 1971.

[BFR98a] P. Bradley, U. Fayyad, and C. Reina. Scaling EM clustering to large databases. Technical Report MSR-TR-98-35, Microsoft Research, Redmond, WA, 1998.

[BFR98b] P. S. Bradley, U. M. Fayyad, and C. Reina. Scaling clustering algorithms to large databases. In *Proc. of 4th Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 9–15, 1998.

[BG98] T. Berger and J. D. Gibson. Lossy source coding. *IEEE Transactions on Information Theory*, 44(6):2691–2723, 1998.

[BG02a] A. Banerjee and J. Ghosh. Frequency sensitive competitive learning for clustering on high-dimensional hyperspheres. In *(To appear) Proc. Intl. Jt. Conf. on Neural Networks (IJCNN)*, May 2002.

[BG02b] A. Banerjee and J. Ghosh. On scaling up balanced clustering algorithms. In *Proc. 2nd SIAM Intl. Conf. on Data Mining*, pages 333–349, April 2002.

[BG04] A. Banerjee and J. Ghosh. Frequency sensitive competitive learning for balanced clustering on high-dimensional hyperspheres. In *IEEE Transactions on Neural Networks*, May 2004.

[BGGM80] A. Buzo, A. H. Gray, R. M. Gray, and J. D. Markel. Speech coding based upon vector quantization. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28(5):562–574, 1980.

[BGW04] A. Banerjee, X. Guo, and H. Wang. Optimal Bregman prediction and Jensen's equality. In *Proc. Intl. Symposium on Information Theory (ISIT)*, 2004.

[BGW05] A. Banerjee, X. Guo, and H. Wang. On the optimality of conditional

expectation as a Bregman predictor. *IEEE Transactions on Information Theory*, 51(7), July 2005. Accepted for publication.

[Bil97] J. Bilmes. A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden markov models. Technical Report ICSI-TR-97-02, University of Berkeley, 1997.

[BL03] A. Blum and J. Langford. PAC-MDL bounds. In *Proc. of 16th Annual Conf. on Computational Learing Theory (COLT)*, 2003.

[BL04] A. Banerjee and J. Langford. An objective evaluation criterion for clustering. In *Proc. 10th Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 515–520, 2004.

[Bla72] R. E. Blahut. Computation of channel capacity and rate-distortion functions. *IEEE Transactions on Information Theory*, 18:460–473, 1972.

[Bli98] J. A. Blimes. A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical report, U. C. Berkeley, April 1998.

[BMDG03] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. Technical Report TR-03-19, Department of Computer Science, University of Texas at Austin, 2003.

[BMDG04] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with bregman divergence. In *Proc. of 4th SIAM Intl. Conf. on Data Mining*, pages 234–245, 2004.

[BN78] O. Barndorff-Nielsen. *Information and Exponential Families in Statistical Theory*. Wiley Publishers, 1978.

[BP92] J. C. Bezdek and S.K. Pal. *Fuzzy Models for Pattern Recognition*. IEEE Press, Piscataway, NJ, 1992.

[Bre67] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Physics*, 7:200–217, 1967.

[BTCT89] A. Ben-Tal, A. Charnes, and M. Teboulle. Entropic means. *Journal of Mathematical Analysis and Applications*, 139:537–551, 1989.

[BYRN99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, New York, 1999.

[CDS01] M. Collins, S. Dasgupta, and R. Schapire. A generalization of principal component analysis to the exponential family. In *Proc. of 14th Annual Conf. on Neural Information Processing Systems (NIPS)*, 2001.

[CG01] K. Chang and J. Ghosh. A unified model for probabilistic principal surfaces. *IEEE Trans. PAMI*, 23(1):22–41, Jan 2001.

[CKPT92] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proc. 15th Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 318–329, 1992.

[CL00] W. Cheney and W. Light. *A Course in Approximation Theory*. Brooks/Cole Publishng Company, 2000.

[Col97] M. Collins. The EM algorithm. Technical report, Department of Computer and Information Science, University of Pennsylvania, 1997.

[Csi67] I. Csisár. Information-type measures of difference of probability distributions and indirect observations. *Studia Sci. Math. Hungar.*, 2:299–318, 1967.

[Csi74] I. Csiszár. On the computation of rate distortion functions. *IEEE Transactions of Information Theory*, IT-20:122:124, 1974.

[Csi91] I. Csiszár. Why least squares and maximum entropy? An axiomatix approach to inference for linear inverse problems. *The Annals of Statistics*, 19(4):2032–2066, 1991.

[Csi95] I. Csiszár. Generalized projections for non-negative functions. *Acta Mathematica Hungarica*, 68(1-2):161–185, 1995.

[CSS00] M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. In *Proc. of 13th Annual Conf. on Computational Learing Theory (COLT)*, pages 158–169, 2000.

[CT91] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

[CZ98] Y. Censor and S. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1998.

[deS88] D. deSieno. Adding conscience to competitive learning. In *IEEE Annual Int'l. Conf. on Neural Networks*, pages 1117–1124, 1988.

[Dev55] A. Devinatz. The representation of functions as Laplace-Stieltjes integrals. *Duke Mathematical Journal*, 24:481–498, 1955.

[DFG01] I. S. Dhillon, J. Fan, and Y. Guan. Efficient clustering of very large document collections. In V. Kumar R. Grossman, C. Kamath and R. Nam-

buru, editors, *Data Mining for Scientific and Engineering Applications*. Kluwer Academic Publishers, 2001.

[DH01] P. Domingos and G. Hulton. A general method for scaling up machine learning algorithms and its application to clustering. In *Proc. 18th Intl. Conf. Machine Learning*, pages 106–113, 2001.

[DHS00] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, 2000.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[DM01] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, 2001.

[DMK03] I. Dhillon, S. Mallela, and R. Kumar. A divisive information-theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research*, 3(4):1265–1287, 2003.

[Dom01] B. E. Dom. An information-theoretic external cluster-validity measure. Technical Report RJ 10219, IBM Research Report, 2001.

[Edw73] C. H. Edwards. *Advanced Calculus of Several Variables*. Academic Press, 1973.

[EGG03] W. Ehm, M. G. Genton, and T. Gneiting. Stationary covariances associated with exponentially convex functions. *Bernoulli*, 9(4):607–615, 2003.

[EKSX96] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A density-based algo-

rithm for discovering clusters in large spatial databses. In *Proc. Intl. Conf. Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[ELL80] B. S. Everitt, S. Landau, and M. Leese. *Cluster Analysis*. Oxford Univ Press, 1980.

[Fas99] D. Fasulo. An analysis of recent work on clustering. Technical report, University of Washington, Seattle, 1999.

[Fel67] W. Feller. *An Introduction to Probability Theory and Its Applications*. John Wiley & Sons, 1967.

[Fis87] D. Fisher. Cobweb: Knowledge acquisition via conceptual clustering. *Machine Learning*, 2:139–172, 1987.

[Fri94] J. H. Friedman. An overview of predictive learning and function approximation. *From Statistics to Neural Networks, Proc. NATO/ASI workshop*, pages 1–61, 1994.

[FS97] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[FW00] J. Forster and M. K. Warmuth. Relative expected instantaneous loss bounds. In *Proc. of 13th Annual Conf. on Computational Learing Theory (COLT)*, pages 90–99, 2000.

[GA96] A. S. Galanopoulos and S. C. Ahalt. Codeword distribution for frequency sensitive competitive learning with one-dimensional input data. *IEEE Trans. Neural Networks*, 7(3):752–756, 1996.

[GD04] P. D. Grünwald and A. Dawid. Game theory, maximum entropy, mini-

mum discrepancy, and robust bayesian decision theory. *Annals of Statistics*, 32(4), 2004.

[GG01] G. K. Gupta and J. Ghosh. Detecting seasonal and divergent trends and visualization for very high dimensional transactional data. In *Proc. 1st SIAM Intl. Conf. on Data Mining*, April 2001.

[Gho03] J. Ghosh. Scalable clustering. In Nong Ye, editor, *The Handbook of Data Mining*, pages 247–277. Lawrence Erlbaum Assoc., 2003.

[GI89] D. Gusfield and R. W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, Cambridge, MA, 1989.

[GJ79] M. R. Garrey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., June 1979.

[GMA97] A. S. Galanopoulos, R. L. Moses, and S. C. Ahalt. Diffusion approximation of frequency sensitive competitive learning. *IEEE Trans. Neural Networks*, 8(5):1026–1030, Sept 1997.

[GMMO00] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams. In *Proc. IEEE Symposium on Foundations of Computer Science*, pages 359–366, 2000.

[GN98] R. M. Gray and D. L. Neuhoff. Quantization. *IEEE Transactions on Information Theory*, 44(6):2325–2383, 1998.

[Gro76] S. Grossberg. Adaptive pattern classification and universal recoding: 1. Parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23:121–134, 1976.

[Gro87] S. Grossberg. Competitive learning: From interactive action to adaptive resonance. *Cognitive Science*, 11:23–63, 1987.

[GRS98] S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. In *In Proc. ACM SIGMOD Intl. Conf.on Management of Data*, pages 73–84, New York, 1998. ACM.

[GRS99] S. Guha, R. Rastogi, and K. Shim. ROCK: a robust clsutering algorithm for categorical attributes. In *Proc. 15th Intl. Conf. on Data Engineering*, 1999.

[GT01] D. Gilbarg and N. Trudinger. *Elliptic Partial Differential Equations of Second Order*. Springer-Verlag, New York, 3nd edition, 2001.

[GV03] P. D. Grünwald and P. Vitányi. Kolmogorov complexity and information theory with an interpretation in terms of questions and answers. *Journal of Logic, Language and Information*, 12(4):497–529, 2003.

[GW93] N.A. Gershenfeld and A.S. Weigend. The future of time series: Learning and understanding. In A. S. Weigend and N.A. Gershenfeld, editors, *Time Series Prediction*, pages 243–264. Addison Wesley, 1993.

[GW00] C. Gentile and M. Warmuth. Proving relative loss bounds for on-line learning algorithms using bregman divergences. In *Proc. of 13th Annual Conf. on Computational Learing Theory (COLT)*, 2000.

[Har17] G. H. Hardy. On stieltjes' "probléme des moments". *Messenger of Mathematics*, 46:175–182, 1917.

[HDR97] G.E. Hinton, P. Dayan, and M. Revow. Modeling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Networks*, 8:65–74, Jan 1997.

[HK98] A. Hinneburg and D. A. Keim. An efficient approach to clustering in large multimedia databses with noise. In *Proc. Intl. Conf. Knowledge Discovery and Data Mining*, pages 58–65, 1998.

[HKT01] J. Han, M. Kamber, and A. K. H. Tung. Spatial clustering methods in data mining: A survey. In *Geographic Data Mining and Knowledge Discovery*. Taylor and Francis, 2001.

[HSD01] G. Hulten, L. Spencer, and P. Domingos. Mining time-changing data streams. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 97–106, San Francisco, CA, 2001. ACM Press.

[HW01] M. Herbster and M. K. Warmuth. Tracking the best linear predictor. *Journal of Machine Learning Research*, 1:281–309, 2001.

[JD88] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, New Jersey, 1988.

[JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[JV01] K. Jain and V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *Journal of the ACM*, 48:274–296, 2001.

[KF75] A. N. Kolmogorov and S. V. Fomin. *Introductory Real Analysis*. Dover Publications Inc., 1975.

[KHK99] G. Karypis, E. H. Han, and V. Kumar. Chameleon: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer*, 32(8):68–75, 1999.

[KK80] D. Kazakos and P.P. Kazakos. Spectral distance measures between Gaussian processes. *IEEE Transactions on Automatic Control*, 25(5):950–959, 1980.

[KK92] J. N. Kapur and H. K. Kesavan. *Entropy Optimization Principles with Applications*. Academic Press, 1992.

[KK98] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.

[KMN97] M. Kearns, Y. Mansour, and A. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *Proc. of 13th Annual Conf. on Uncertainty in Artificial Intelligence (UAI)*, pages 282–293, 1997.

[Kni94] O. Knill. Probability. Course notes from Caltech, 1994.

[KR90] L. Kaufman and P. J. Rousseeuw. *Finding groups in data: an Introduction to cluster analysis*. John Wiley & Sons, 1990.

[KT74] S. Karlin and H. M. Taylor. *A First Course in Stochastic Processes*. Academic Press, 2nd edition, 1974.

[KW01] J. Kivinen and M. K. Warmuth. Relative loss bounds for multidimensional regression problems. *Machine Learning*, 45:301–329, 2001.

[LBG80] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95, 1980.

[LC98] E. L. Lehmann and G. Casella. *Theory of Point Estimation*. Springer, 2nd edition, 1998.

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. Math. Statist. Prob.*, volume 1, pages 281–297, 1967.

[Mar75] K. V. Mardia. Statistics of directional data. *J. Royal Statistical Society. Series B (Methodological)*, 37(3):349–393, 1975.

[McL55] N. W. McLachlan. *Bessel Functions for Engineers*. Oxford University Press, 1955.

[Mei03] M. Meilă. Comparing clusterings by the variation of information. In *Proc. of 16th Annual Conf. on Computational Learing Theory (COLT)*, 2003.

[MH98] M. Meila and D. Heckerman. An experimental comparison of several clustering and initialization methods. Technical Report MSR-TR-98-06, Microsoft Research, 1998.

[Mit97] T. M. Mitchell. *Machine Learning*. McGraw-Hill Intl, 1997.

[MK96] G. J. McLachlan and T. Krishnan. *The EM algorithm and Extensions*. Wiley-Interscience, 1996.

[MN89] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman & Hall/CRC, 1989.

[Mov] Grouplens research. http://www.cs.umn.edu/Research/GroupLens/index.html.

[Moz93] M. C. Mozer. Neural network architectures for temporal sequence processing. In A. S. Weigend and N.A. Gershenfeld, editors, *Time Series Prediction*, pages 243–264. Addison Wesley, 1993.

[MR95] R. Motwani and P. Raghavan. *Randmized Algorithms*. Cambridge University Press, 1995.

[MS03] D. Modha and S. Spangler. Feature weighting in k-means clustering. *Machine Learning*, 52(3):217–237, 2003.

[N20] The 20 newsgroup dataset. http://people.csail.mit.edu/people/jrennie/20Newsgroups/.

[NH98] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. MIT Press, 1998.

[Nig01] K. P. Nigam. *Using Unlabeled Data to Improve Text Classification*. PhD thesis, School of Computer Science, Carnegie Mellon University, May 2001.

[NSF] NSF research award abstracts. http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html.

[Pal97] M. Palus. On entropy rates of dynamical systems and Gaussian processes. *Physics Letters A*, 227(5-6):301–308, 1997.

[Pap91] A. Papoulis. *Probability, Random Variables and Stochastic Processes*. McGraw Hill, 1991.

[PF99] C. R. Palmer and C. Faloutsos. Density biased sampling: An improved method for data mining and clustering. Technical report, Carnegie Mellon University, May 1999.

[PKC94] J. C. Principe, J.-M. Kuo, and S. Celebi. An analysis of the gamma memory in dynamic neural networks. *IEEE Transactions on Neural Networks*, 5:331–337, March 1994.

[RG98] V. Ramamurti and J. Ghosh. On the use of localized gating in mixtures of experts networks. In *(invited paper), SPIE Conf. on Applications and Science of Computational Intelligence, SPIE Proc. Vol. 3390*, pages 24–35, Orlando, Fl., April 1998.

[RG99] V. Ramamurti and J. Ghosh. Structurally adaptive modular networks for nonstationary environments. *IEEE Transactions on Neural Networks*, 10(1):152–160, 1999.

[Roc70] R. T. Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics. Princeton University Press, 1970.

[Ros94] K. Rose. A mapping approach to rate-distortion computation and analysis. *IEEE Transactions on Information Theory*, 40(6):1939–1952, 1994.

[Ros98] K. Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of IEEE*, 86(11):2210–2239, 1998.

[RW84] R. Redner and H. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, 1984.

[RZ85] D. E. Rumelhart and D. Zipser. Feature discovery by competive learning. *Cognitive Science*, 9:75–112, 1985.

[SCZ98] G. Sheikholesami, S. Chatterjee, and A. Zhang. Wavecluster: A mutiresolution clustering approach for very large spatial databases. In *Proc. Intl. Conf. Very Large Data Bases*, pages 428–439, 1998.

[SDN87] J. P. Changeux S. Dehaene and J. P. Nadal. Neural networks that learn temporal sequences by selection. *Proc. National Academy of Sciences, USA*, 84:2727–2731, May 1987.

[SG97] B. W. Stiles and J. Ghosh. A habituation based neural network for spatio-temporal classification. *Neurocomputing*, 15:273–303, July 1997.

[SG00] A. Strehl and J. Ghosh. A scalable approach to balanced, high-dimensional clustering of market-baskets. In *Proc 7th Intl Conf on High Performance Computing (HiPC 2000)*, December 2000.

[SG02] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research (in review)*, 2002.

[SGM00] A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Proc 7th Natl Conf on Artificial Intelligence : Workshop of AI for Web Search (AAAI 2000)*, pages 58–64. AAAI, July 2000.

[SK01] J. Sinkkonen and S. Kaski. Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14:217–239, 2001.

[ST43] J. A. Shohat and J. D. Tamarkin. *The Problem of Moments*. American Mathematical Society, 1943.

[SW02] N. Slonim and Y. Weiss. Maximum likelihood and the information bottleneck. In *Proc. 16th Annual Conf. on Neural Information Processing Systems (NIPS)*, 2002.

[TNLH01] A. K. H. Tung, R. T. Ng, L. V. S. Laksmanan, and J. Han. Constraint-based clustering in large databses. In *Proc. Intl. Conf. on Database Theory (ICDT'01)*, Jan 2001.

[TPB99] N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Proc. of the 37th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[Tra91] H. G. C. Traven. A neural network approach to statistical pattern classification by "semiparametric" estimation of probability density functions. *IEEE Transactions on Neural Networks*, 2(3):366–377, 1991.

[Tre68] H. L. Van Trees. *Detection, Estimation and Modulation Theory (Part I)*. John Wiley & Sons, 1968.

[Tur96] P.D Turney. The management of context-sensitive features: A review of strategies. In *Proceedings of the ICML-96 Workshop on Learning in Context-Sensitive Domains*, Bari, Italy, July 1996.

[Val64] F. A. Valentine. *Convex Sets*. McGraw-Hill, 1964.

[Wil91] D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.

[WJ03] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical Report TR 649, Dept. of Statistics, University of California at Berkeley, 2003.

[WL90] B. Widrow and M. A. Lehr. 30 years of adaptive neural networks: Perceptron, Madaline and Backpropagation. *Proc. IEEE*, 78(9):1415–1442, Sept 1990.

[WS85] B. Widrow and S.D. Stearns. *Adaptive Signal Processing*. Prentice-Hall, Englewood Cliffs, N.J., 1985.

[WYM97] W. Wang, J. Yang, and R. Muntz. Sting: A statistical information grid approach to spatial data mining. In *Proc. Intl. Conf. Very Large Data Bases*, pages 186–195, 1997.

[ZL02] Y. J. Zhang and Z. Q. Liu. Self-splitting competitive learning: A new

on-line clustering paradigm. *IEEE Transactions on Neural Networks*, 13(2):369–380, March 2002.

[ZRL96] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An efficient data clustering method for very large databases. In *In Proc. ACM SIGMOD Intl. Conf. on Management of Data*, pages 103–114, Montreal, 1996. ACM.

# Vita

Arindam Banerjee was born in Burdwan, India. He received his Bachelor of Engineering degree in Electronics and Tele-communication Engineering from Jadavpur University, India, in Summer 1997 and Master of Technology degree from the Indian Institute of Technology, Kanpur, India, in Summer 1999. He is currently pursuing his Ph.D. in the Computer Engineering area of the Department of Electrical and Computer Engineering at the University of Texas at Austin. He has received several awards, including the Best Algorithms Paper Award at the SIAM International Conference on Data Mining in 2004 and the IBM PhD fellowship for the academic years 2003-2004 and 2004-2005.

Permanent Address: F-4, Tarabag, Burdwan,
West Bengal, 713104, India

This dissertation was typeset with LaTeX $2_\varepsilon$[1] by the author.

[1] LaTeX $2_\varepsilon$ is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin, and extended by Bert Kay, James A. Bednar, and Ayman El-Khashab.